REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	**2. REPORT TYPE**	**3. DATES COVERED** *(From - To)*
17 June 2016	Conference Paper with Briefing Charts	17 June 2016 – 27 July 2016

4. TITLE AND SUBTITLE
SM/MURF: Current Capabilities and Verification as a Replacement of AFRL Plume Simulation Tool COLISEUM (Conference Paper with Briefing Charts)

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)
Samuel J. Araki, Robert S. Martin, David L. Bilyeu, Justin W. Koo

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER
Q1NC

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) AND ADDRESS(ES)

Air Force Research Laboratory (AFMC)
AFRL/RQRS
1 Ara Drive
Edwards AFB, CA 93524-7013

8. PERFORMING ORGANIZATION REPORT NO.

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Air Force Research Laboratory (AFMC)
AFRL/RQR
5 Pollux Drive
Edwards AFB, CA 93524-7048

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)
AFRL-RQ-ED-TP-2016-156

12. DISTRIBUTION / AVAILABILITY STATEMENT
Approved for Public Release; Distribution Unlimited. PA Clearance Number: 16317 Clearance Date: 7/27/2016
The U.S. Government is joint author of the work and has the right to use, modify, reproduce, release, perform, display, or disclose the work.

13. SUPPLEMENTARY NOTES
For presentation at 52nd AIAA/SAE/ASEE Joint Propulsion Conference; Salt Lake City, Utah July 25-27, 2016
Prepared in collaboration with ERC (Conference Paper with Briefing Charts)

14. ABSTRACT

The Spacecraft Multi-Scale/Multi-Physics Universal Research Framework (SM/MURF) is currently in development at the in-space propulsion branch of the Air Force Research Laboratory (AFRL). This framework unifies multiple research codes developed independently at this branch and is intended to simulate plasma under a wide range of time and length scales relevant to the spacecraft electric propulsion (EP) systems. The current development effort focuses on the modernization of the AFRL plume simulation tool, COLISEUM. In this paper, all the operations for a plume simulation are reviewed along with the results from integration tests to verify the correctness of those operations. As the final verification test, SM/MURF and COLISEUM are used to perform the same baseline plume simulation, and the results from the two codes are compared. In general, the agreement between the two codes is very good. A slight discrepancy was caused by a mismatch of where the field data are computed; SM/MURF and COLISEUM compute field data on cell-centers and nodes, respectively. This results in a slightly different electric field, cascading through particle trajectories, field calculations, sputter rate, and redeposition rate onto spacecraft components.

15. SUBJECT TERMS
N/A

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON**
					J. Koo
a. REPORT	**b. ABSTRACT**	**c. THIS PAGE**	SAR	53	**19b. TELEPHONE NO** *(include area code)*
Unclassified	**Unclassified**	**Unclassified**			N/A

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std. 239.18

SM/MURF: Current Capabilities and Verification as a Replacement of AFRL Plume Simulation Tool COLISEUM

Samuel J. Araki*, Robert S. Martin*, David L. Bilyeu†, and Justin W. Koo†

ERC Inc., In-Space Propulsion Branch,†*

Air Force Research Laboratory, Edwards Air Force Base, CA 93524, USA

The Spacecraft Multi-Scale/Multi-Physics Universal Research Framework (SM/MURF) is currently in development at the in-space propulsion branch of the Air Force Research Laboratory (AFRL). This framework unifies multiple research codes developed independently at this branch and is intended to simulate plasma under a wide range of time and length scales relevant to the spacecraft electric propulsion (EP) systems. The current development effort focuses on the modernization of the AFRL plume simulation tool, COLISEUM.

In this paper, all the operations for a plume simulation are reviewed along with the results from integration tests to verify the correctness of those operations. As the final verification test, SM/MURF and COLISEUM are used to perform the same baseline plume simulation, and the results from the two codes are compared. In general, the agreement between the two codes is very good. A slight discrepancy was caused by a mismatch of where the field data are computed; SM/MURF and COLISEUM compute field data on cell-centers and nodes, respectively. This results in a slightly different electric field, cascading through particle trajectories, field calculations, sputter rate, and redeposition rate onto spacecraft components.

I. Introduction

The Spacecraft Multi-Scale/Multi-Physics Universal Research Framework (SM/MURF) is currently in development at the in-space propulsion branch of the Air Force Research Laboratory (AFRL/RQRS); this framework is capable of simulating plasma and gas under a wide range of time and length scales relevant to the spacecraft electric propulsion systems. These conditions range from a high density plasma such as the one seen in the Field Reversed Configuration (FRC) thrusters to the relatively low density plasma found in the Hall and ion thrusters. To simulate these different conditions, separate operators are implemented in SM/MURF, including Magnetohydrodynamics (MHD) with detailed kinetic chemical reactions, Vlasov solvers, and a Particle-In-Cell (PIC) solver with both Direct Simulation Monte Carlo (DSMC) and Monte Carlo Collision (MCC) methods. Although these solvers have very different computational requirements, they are all unified under SM/MURF. SM/MURF is export controlled under ITAR but is build upon the non-ITAR infrastructure, Thermophysics Universal Research Framework (TURF)[1] also in development at AFRL/RQRS. As such, the two codes share the same coding standards, interfaces, and techniques.

The unification of the SM/MURF operators is particularly useful in bridging different numerical models. For example, a full PIC simulation of a plasma can easily be reduced to a hybrid fluid/PIC simulation by changing descriptions of heavy species as fluids through a small alteration of the input file. The framework also enables a more rigorous comparison of models. For example, one can again change a block of input file to swap models while retaining the other parts the same. In this way, it is ensured that the simulation is setup in exactly the same manner. Furthermore, SM/MURF can be easily expanded by adding operations underneath the framework. In other words, a numerical research can be conducted on top of this framework, allowing other researchers to more easily maintain or take over the research.

*Computational Scientist, ERC Inc., 4 Draco Drive, Edwards AFB, CA 93524

†Computational Scientist, AFRL/RQRS, 4 Draco Drive, Edwards AFB, CA 93524

The first release version of SM/MURF contains the functionalities required for flight support simulations performed by the AFRL plume simulation tool, COLISEUM.[2–5] These include an unstructured surface mesh loader, particle-surface interactions, calculations of flow and surface fields, visualizations of fields, an accurate scattering calculation of high energy ions with a background gas, Poisson and Boltzmann potential solvers, numerical probes, and source models for particle injections. The objective of this paper is to review the current state of development along with the results from the integration tests for these functions. This paper will also present the comparison of the results from a baseline plume simulation by SM/MURF and COLISEUM.

II. SM/MURF Framework

The most recent paradigm shift in computing architecture occurred with an introduction of co-processor technology such as Graphical Processing Unit (GPU) and Intel Many Integrated Core (MIC) Architecture. This shift has enabled a further increase in computing power, roughly maintaining a trend according to Moore's Law where computing power doubles every 24-30 months. Among the "Top500" list of the most powerful supercomputers, a total of 90 systems use the co-processor technology as of June 2015. These heterogeneous systems utilize both the multi-core CPUs and co-processors, and are addressed through combinations of the Message Passing Interface (MPI), Open Multi-Processing (OpenMP) compiler directives, and NVIDIA's Compute Unified Device Architecture (CUDA).

When COLISEUM began its development, the primary parallelization strategy was to use domain decomposition and MPI. Therefore, a major rewrite of the code for a hybrid "MPI+OpenMP/GPU" architecture was necessary to utilize the full capabilities of supercomputers with the heterogeneous architecture. Furthermore, the COLISEUM's domain decomposition strategy was problematic in obtaining a good scaling with number of CPU cores. COLISEUM statically partitions a computational domain and distributes each sub-domain to a CPU core. However, the number of operations heavily depends on the number of particles, which is often not distributed nearly equally across sub-domains in a realistic problem. These have motivated a development of a new framework.

In SM/MURF, a CPU core can perform calculations for multiple sub-domains; this strategy has been chosen to enable dynamic load balancing between CPUs. Every CPU has its own "World" object that contains global information about the simulation as well as the list of sub-domains. Then, every sub-domain contains list of objects (i.e. particle distributions, field, and operations) in addition to its size and index. This tree-hierarchy structure is a key element of the framework, which is attained by the General Service Object (GSObject) or a branching double-linked list with member functions as shown in Fig. 1. A GSObject includes "name" and "type-name" information to enable high-level search and access routine. The key functionality of GSObject is serialization/deserialization routine that help unify the memory transfer between host and GPU as well as between MPI processes. In SM/MURF, most of class objects are inherited from GSObject to form the tree-hierarchy structure with the World object as a common ancestor (see Fig. 2)

When partitioning the computational domain, layers of ghost cells are added to sub-domains. These ghost cells extend to neighboring sub-domains and are used to patch information across sub-domains as well as to simplify a treatment of boundary conditions. Communication between sub-domains is handled by patch objects that are members of World. The patch can exchange data between domains directly within a process, while the patch sends and receives the data over the network across MPI processes. SM/MURF currently performs several patch operations for field data such as `MAX`, `MIN`, `SUM`, and `PRODUCT`. During every communication stage, the patch operation is performed, and computed value is stored in the non-ghost cell while initializing the ghost cells to be zero. The patching of particles is done by first splitting the particles in ghost cells to a separate distribution, and merging the distribution with the one in the neighboring sub-domain.

In SM/MURF, data operations have been distilled down to a standard operation interface, including an "init" routine, an "apply" routine, and one or more "core" routines. In the "init" routine, generic parameters are stored in a standard template library (STL) map structure that is passed to the individual operations "init" routine where it is interpreted. Once constructed, the operations are stored in vector stacks generically understood to contain a list of operations. The primary loop of the code cycles through this stack invoking the "apply" routine to each operation in turn. This function uses a set of inputs and outputs pre-defined in the initialization to advance the state of the simulation. Within the "apply" routine, the "core" routine is invoked in parallel on pieces of independent data to maximize parallelism. These cores can run on a

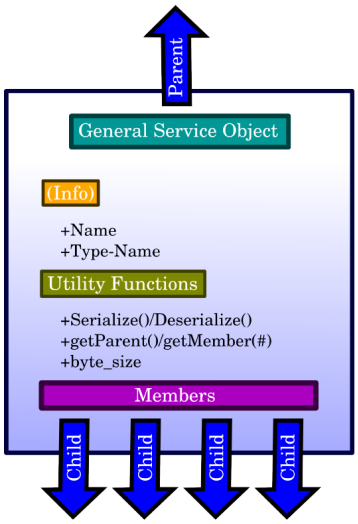


Figure 1. Schematic of GSObject.

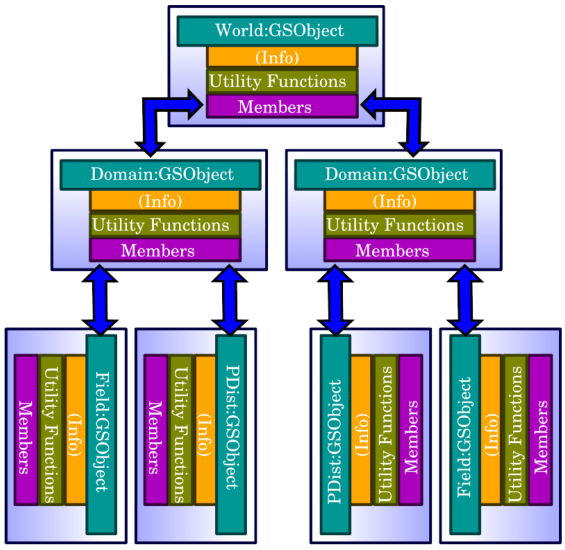


Figure 2. Tree-hierarchy structure of objects with GSObject.

single CPU core, multiple CPU cores via OpenMP, or on an NVIDIA GPU via CUDA. In order to facilitate multiple communication patterns, the operation stacks are broken into"stages." Stages can be subcycled as necessary with conditions placed on advancing to the next stage.

III. Structure of Baseline Plume Simulations

A SM/MURF simulation starts by loading global inputs from a world input file, *world.list*, and setting up the World object for every MPI process. These inputs include a file name that has a list of operations, a time-step, an end time of a simulation, field data names, available materials, and stage names. It also contains domain information such as domain names, lower and upper bounds of the domain boxes, and how they are partitioned. Every World object creates all the sub-domains that cover the entire simulation domain, but the ones not used within a MPI process are deactivated. Then, a Cartesian structured mesh, SMesh, and field data are initialized within the active sub-domains. After the sub-domains are set, operations defined in an operation list file are added to the World object, and parameters defined within the same block are stored in a STL map structure to be accessed during the operation's init routine. A list of operations that are used in plume simulations is provided in table 1.

The init routines not only use input parameters to set local variables within the operation class objects but also perform operations that need to be done once before a simulation starts. For example, a plume simulation requires a spacecraft geometry to be loaded as a unstructured surface mesh, UMesh (`LogicalUMeshImporterOp`). SM/MURF then needs to figure out how UMesh is immersed in the region of SMesh (`LogicalMeshSurfaceSugarcubeOp`). With rigorous geometric algorithms, every cell has a list of UMesh triangular elements that intersect the cell and knows if it is in a free space, within a geometry, or at the geometry boundary. This information is required when applying Dirichlet boundary condition for an electric potential and setting it to a constant value within a spacecraft component (`LogicalPotentialSetInsideGeometryOp`). Also, the intersecting UMesh element list is used to find particle-surface intersections and apply surface interaction operations. The init routines also create multiple particle distributions that hold particle's position, velocity, species, cell index, tag, etc (`MSPDistInitOp`). Furthermore, a UMesh field data object is created at this time (`UFieldMSInitOp`). In order to identify regions with respect to the geometry consistently for different number of sub-domains, contiguous regions are unified across all the sub-domains (`LogicalMeshGlobalRecolorOp`). Furthermore, one MPI process is spawned to run a Hall thruster program called HPHall to obtain a realistic particle distribution from the thruster (`MSPDistSpawnHPHallOp`).

The apply routines are executed in the order specified in the operations file, and a MPI communication is done between stages to exchange data between MPI processes. Figure 3 shows a simplified flowchart of

Table 1. List of operations for SM/MURF plume simulations.

Operation	Description
`CriteriaStageOp`	Goes to next stage when a criterion is met.
`GSOPatchOp`	Patchs GSObject across domain. Used to patch particles.
`LogicalMeshSurfaceSugarcubeOp`	Determines relations between an object represented by a unstructured surface mesh (UMesh) and a structured mesh (SMesh).
`LogicalMeshGlobalRecolorOp`	Unifies indices of the same regions set differently across sub-domains.
`LogicalNodeGradientOp`	Computes a gradient at a node from 8 neighboring cell-centers.
`LogicalFieldPatchOp`	Sets patch object for SMesh field data.
`LogicalFieldScalarMulOp`	Applies scaler operations to field data
`LogicalFieldSetOp`	Sets field data to a value.
`LogicalFieldTimeAverageOp`	Takes a time average of field data.
`LogicalFieldVolumetricMulOp`	Multiplies or divides field data by cell volumes.
`LogicalFieldWriteVTKOp`	Writes 3D field data to a file in VTK format.
`LogicalFieldWriteVTK2DOp`	Writes field data along a 2D plane to a file in VTK format.
`LogicalPotentialSetInsideGeometryOp`	Assigns a value to an integer field that determines if inside or outside the body and which potential solver to use.
`LogicalPotentialBoltzmannOp`	Solves electric potential from Boltzmann relation
`LogicalUMeshImporterOp`	Imports a unstructured mesh.
`MSPDistCellIDOp`	Assigns cell ID to a particle distribution. Sets to the maximum cell number if outside the sub-domain.
`MSPDistChargeDepositionOp`	Performs zeroth or first order charge deposition to SMesh.
`MSPDistCombineOp`	Combines two particle distributions.
`MSPDistCopyOp`	Copies a particle distribution to another distribution.
`MSPDistEmptyOp`	Empties a particle distribution by setting the number of particles to be zero.
`MSPDistESPushOp`	Performs an explicit electrostatic push to a particle distribution.
`MSPDistInitOp`	Creates a particle distribution.
`MSPDistMCCElasticFitOp`	Applies elastic collisions to particles by sampling from a differential cross-section.
`MSPDistNormalMaxwellianStreamOp`	Injects particles according to a streaming Maxwellian distribution.
`MSPDistParticleCountOp`	Counts number of particles in a distribution.
`MSPDistProbeFixedOp`	Samples current to a probe attached to a surface mesh component.
`MSPDistProbeStageSphericalOp`	Samples current to a virtual spherical probe.
`MSPDistReadVTKOp`	Reads a particle distribution data from a VTK file for restart.
`MSPDistRemovePartOp`	Removes particles from a distribution according to their cell ID or weight.
`MSPDistSampleOp`	Samples SMesh field data from a particle distribution.
`MSPDistSourceRPAOp`	Reads a RPA data in csv format and uses the data to inject particles.
`MSPDistSpawnHPHallOp`	Spawns a MPI process and runs HPHall as a particle source. Collects particles at every time-step.
`MSPDistSplitOp`	Splits a particle distribution to two according to particles cell ID.
`MSPDistSortOp`	Sorts particles in the order of cell ID. particles with cell IDs larger than the maximum are thrown.
`MSPDistSugarcubeSurfIntersectionOp`	Determines if particles intersect with UMesh elements.
`MSPDistSurfaceInteractionOp`	Performs sputtering and surface reflection calculations.
`MSPDistWriteVTKOp`	Writes a particle distribution data to a file in VTK format.
`NextStageOp`	Goes to the next stage.
`ProbeFixedWriteOp`	Writes a fixed probe data to a csv file.
`ProbeStageWriteOp`	Writes a spherical probe data to a csv file.
`SampleFieldReadVTKOp`	Reads a sampled field data in VTK format for restart.
`SampleFieldWriteVTKOp`	Writes a sampled field data to a file in VTK format for restart.
`UFieldMSComputeOp`	Computes UMesh field data.
`UFieldMSInitOp`	Creates UMesh field object.
`UFieldWriteVTKOp`	Writes UMesh field data to a file in VTK format.

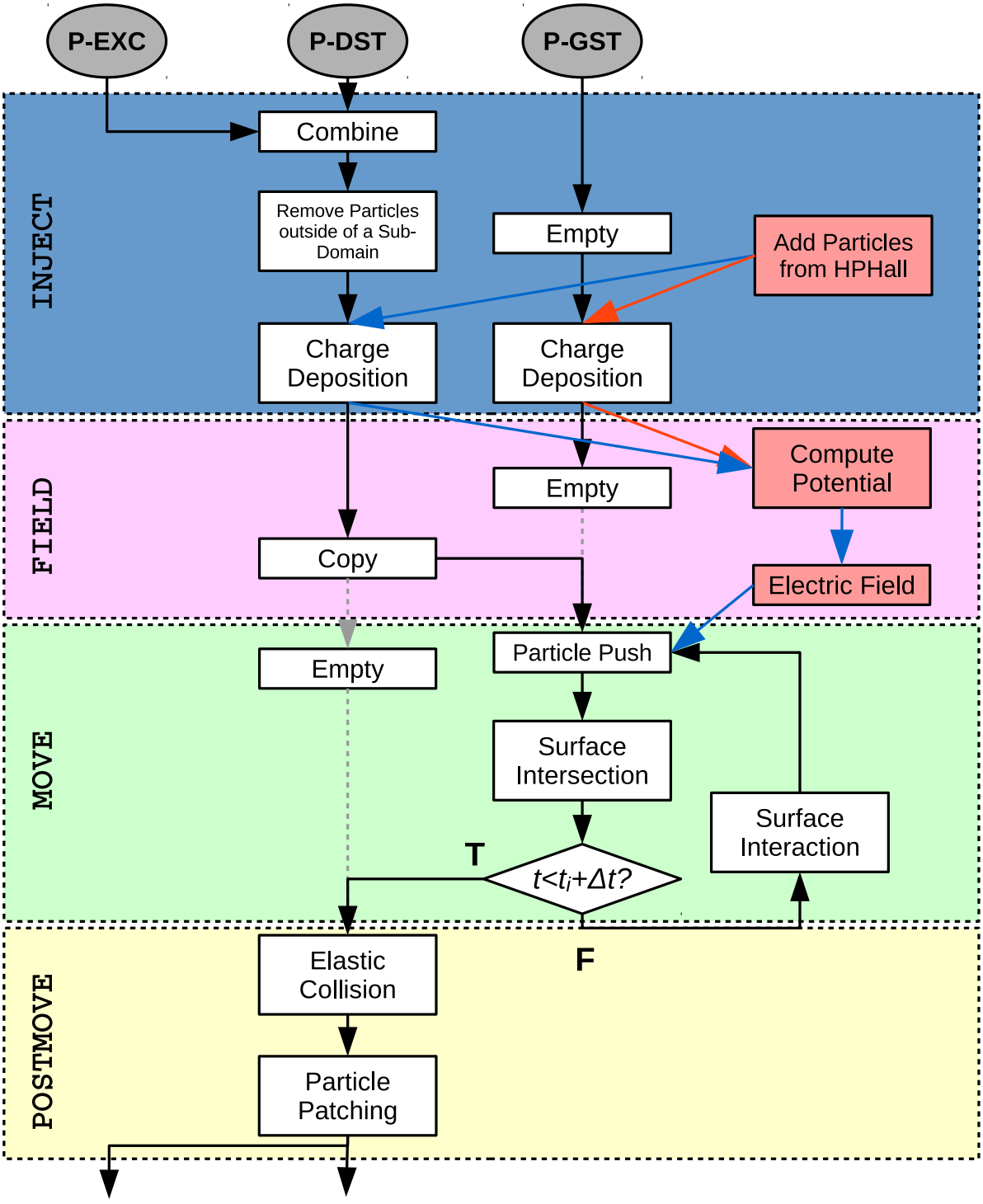


Figure 3. Simplified flowchart for SM/MURF plume simulations. `P-DST`, `P-GST`, and `P-EXC` are the particle distributions.

a SM/MURF plume simulation. Four separate stages are used in this example; they are `INJECT`, `FIELD`, `MOVE`, and `POSTMOVE`. More stages can be added but may slow down the simulation due to the increased number of MPI communications. In the `INJECT` stage, particles are injected into the simulation domain (`MSPDistSpawnHPHallOp`), and their charges are distributed to the SMesh cells (`MSPDistChargeDeposition-Op`). In the `FIELD` stage, electric potential is computed based on the charge density (`LogicalPotential-BoltzmannOp`), and the cell-centered potential values are used to compute electric fields at the grid nodes (`LogicalNodeGradientOp`). The charge deposition operation has to be performed before the `FIELD` stage in order to patch the charge density values when the first-order weighting scheme is used. The `MOVE` stage involves explicit particle advancement in time (`MSPDistESPushOp`). This stage has an extra loop to account for particles not reaching the final time within one iteration as they impact surfaces. Finally, the `POSTMOVE` stage performs a collision calculation (`MSPDistMCCElasticFitOp`) and splits particles that are outside of a sub-domain to a different particle distribution (`MSPDistSplitOp`). These particles are patched to other sub-domain (`GSOPatchOp`) or thrown away if outside of the whole domain in the `INJECT` stage (`MSPDistRemovePartOp`). Other additional operations that are not shown in Fig. 3 can be found in table 1.

IV. Operations and Integration Tests

SM/MURF utilizes a testing framework to ensure consistent quality during development. To accomplish this, tests are designed to check both individual functions (unit tests) as well as how multiple operations work together (integration tests). The statistical nature of PIC codes makes this particularly difficult. These tests require running the simulation for a long time to generate a statistical average which has to be close enough to a known value. An alternative would be provide a known set of "random" values and ensure that the results remain the same. The downside of this approach is that the test does not reflect how the simulation is used in practice. Currently, integration tests are written for Maxwellian source (A.1), electrostatic particle push (D), numerical probes (F), particle sampling to SMesh (G), elastic collision (H), particle sampling to UMesh (I), particle and SMesh field patching across sub-domains (J and K), and particle distribution and sampling field read and write (L and M). These unit and integration tests are set to run periodically on our server machine through an open source continuous integration tool, Jenkins. Jenkins first pulls a SM/MURF repository, builds, run unit tests, run integration tests, and finally output a summery of the regression test. Jenkins requires a xml format output for tests, so the unit tests are written in c++ with google test suite, and integration tests are written in python with unittest and xmlrunner modules.

A. Source Models

1. Streaming Maxwellian

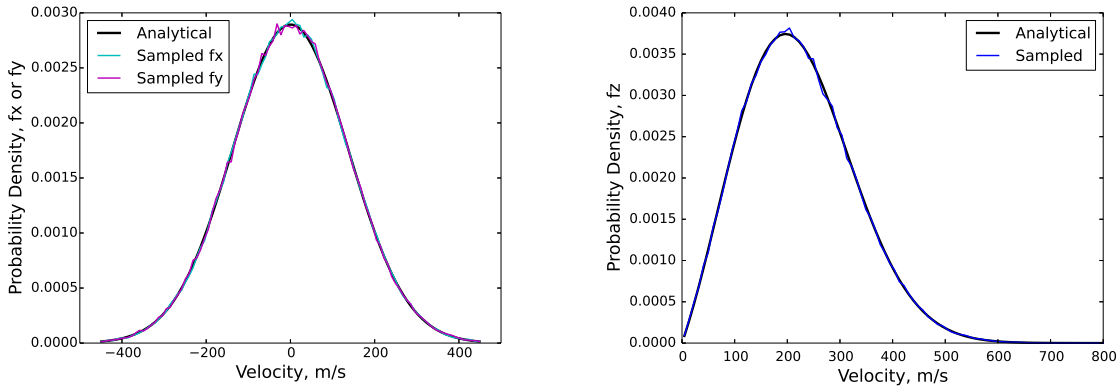
The streaming Maxwellian source (`MSPDistNormalMaxwellianStreamOp`) injects particles according to a one-sided Maxwellian distribution with a streaming velocity normal to an injection surface. This source does not provide a realistic distribution for charged species at a thruster exit but is generally good for neutral particles. The operation determines the number of particles to generate from a given surface and samples particles from distribution functions. The distribution function for a tangent velocity is given as

$$f_x = \frac{\beta}{\sqrt{\pi}} \exp\left(-\beta^2 v_x^2\right) \tag{1}$$

where β is an inverse of thermal velocity, i.e. $\beta = m/(2k_B T)$, m is the particle mass, k_B is the Boltzmann constant, T is the temperature, and v is the particle velocity. The distribution function in the normal direction is modified due to the streaming velocity.[6]

$$f_z = \frac{2\beta^2 v \exp\left(-\beta^2 (v - v_n)^2\right)}{s_n \sqrt{\pi}(1 + \mathrm{erf}(s_n)) + \exp(-s_n^2)} \tag{2}$$

where v_n is the streaming velocity and s_n is the speed ratio of v_n to the thermal velocity, $s_n = \beta v_n$. In this operation, tangential velocities are determined by Box-Muller algorithm[7] and normal velocity is determined with cumulative distribution function of Eq. (2) and applying acceptance-rejection routine to the distribution function.[8]



(a) Tangent direction

(b) Normal direction

Figure 4. Analytical and sampled distribution functions of the Maxwellian source.

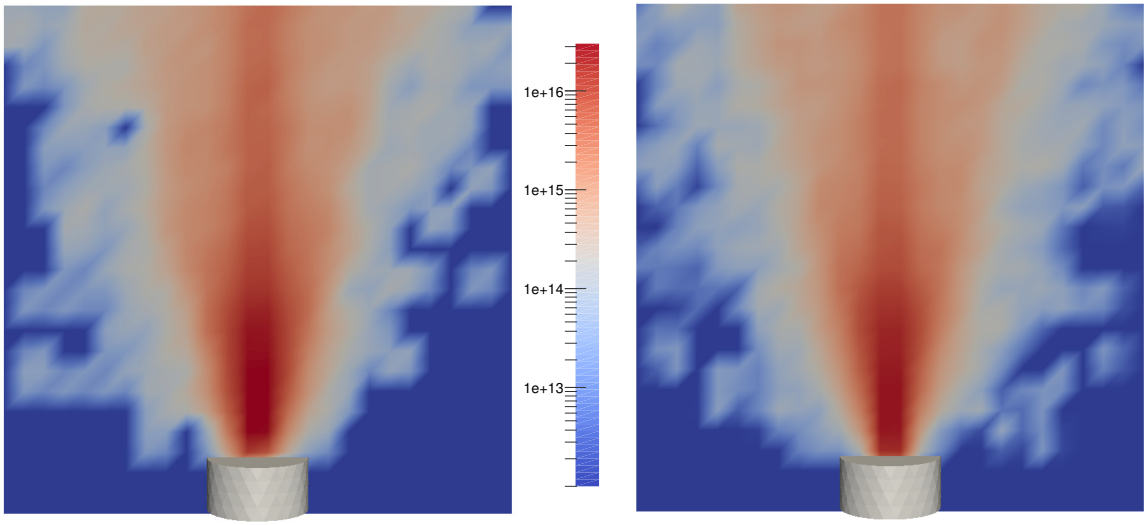
The implementation can be verified by sampling particles right after injecting particles by the Maxwellian source. The post-injection velocity distribution must be close to the distributions sampled from. The integration test compares the analytical distributions with the sampled distributions and checks if the maximum norm is within some bound. Figure 4 shows the distribution functions for a Maxwellian source injecting at $T = 300$ K and $v_n = 200$ m/s. The distribution functions are in a good agreement with the analytical distributions.

2. HPHall Source

HPHall is a hybrid fluid/PIC code, originally developed by MIT in 1998,[9] to simulate a time-dependent evolution of plasma in a Hall effect thruster (HET). Multiple versions of HPHall exist in the community, and the version at AFRL is branched from a version at Jet Propulsion Laboratory in 2008. Since then, only a slight modification has been made to improve the interface with COLISEUM. COLISEUM spawns one MPI process to launch HPHall and gathers particles of different species (singly and double charged ions and neutral atoms) that are located downstream of the thruster exit at every time-step. The charged particles are weighted to the grid to compute a charge density. Then, particles located too close to the thruster exit are thrown, and only the particles passing a certain region away from the thruster exit are tracked in COLISEUM. Once HPHall parameters are tuned to match global performance data of the specific Hall thruster, the HPHall source can provide a more realistic particle distribution compared to other particle source models. The same operation has been implemented in SM/MURF (`MSPDistSpawnHPHallOp`). Figure 5 shows the ion densities from SM/MURF and COLISEUM after 0.2 ms. In this example, potential and collision calculations are turned off to eliminate other factors that can contribute to the difference. The outputs from the two codes generally agree well, and the difference is primarily attributed to where the field data are computed; COLISEUM computes field data at computational nodes while SM/MURF computes the cell-centered field data. As a result, the data in under-resolved regions can be shifted by half a cell. The difference is expected to be smaller as the grid resolution is increased.

3. Retarding Potential Analyzer Source

The Retarding Potential Analyzer (RPA) source (`MSPDistSourceRPAOp`) is designed to take a user-defined energy distribution curve and use it as a source. The energy distribution curve can be determined experimentally by placing RPA probes in different locations around the thruster. This type of source is useful when trying to match simulation data against ground based measurements or when a full hall effect simulation is not possible. SM/MURF also has the capability to mimic a RPA probe through the use of (`MSPDistProbeStageSphericalOp`).



(a) SM/MURF (b) COLISEUM

Figure 5. Instantaneous ion densities after 0.2 ms from two different models with the same particle source.

B. Charge Deposition

Before computing an electric potential, particle charges are weighted to SMesh, and charge densities are computed by dividing the charges by cell volumes. SM/MURF is currently capable of performing both the zeroth and first order weighting schemes (`MSPDistChargeDepositionOp`). The zeroth order weighting is done by simply accumulating the particle charges within a cell.

$$Q_i = \sum_{p}^{N_i} q_p \tag{3}$$

where i is the cell index, p is the particle index, $N_{p,i}$ is the total number of particles within the given cell, and q is the particle charge. For an one-dimensional problem, the first order wighting distributes particle charges to two neighboring cells.

$$Q_i = \sum_{p}^{N_{i-1}+N_i+N_{i+1}} q_p S(x_p) \tag{4}$$

where S is the first order shape function given as

$$S(x_p) = \begin{cases} 0 & x_p < x_i - \frac{1}{2}\Delta x \\ \frac{x_p - x_{i-1}}{x_i - x_{i-1}} & x_i - \frac{1}{2}\Delta x \leq x_p < x_i \\ \frac{x_{i+1} - x_p}{x_{i+1} - x_i} & x_i \leq x_p < x_i + \frac{1}{2}\Delta x \\ 0 & x_i + \frac{1}{2}\Delta x < x_p \end{cases} \tag{5}$$

Equation (5) can easily be extended to three dimensions such that a particle charge is distributed to eight neighboring cells. When using the first order method, charges can be distributed to the first layer of ghost cells. For a multi-domain simulation, this can cause inaccurate results near the domain boundaries if charge density patching across sub-domains is not performed before electric potential calculations. Simple unit tests for both weighting schemes are implemented in SM/MURF, in which charged particles are placed at known locations, and the test checks if the accumulated charges equal to the expected values. The charge density patching is tested in an integration test for SMesh field patch described in Sec. IV.K.

C. Potential Solver

SM/MURF offers two options to calculate the electric potential. The first method solves the Poisson's equation while the second method uses the Boltzmann relation. Solving Poisson's equation is the preferred

method but requires that the grid spacing is less than or equal to the local Debye length. This resolution is not an issue when simulating the near regions of an EP thruster but becomes more computationally expensive as the domain size increases. However, using such a small resolution causes the number of cells to be excessively large and can be difficult to perform a simulation in a reasonable amount of time. For this reason, the electric potential can also be computed from the Boltzmann relation when the grid size is much larger than the Debye length. The underlying assumption here is that the plasma is quasi-neutral within a given cell and that an electron fluid has reached equilibrium well within a time-step. For a plume simulation in a Low Earth Orbit (LEO) environment, the Debye length is typically smaller than the cell size, and the Boltzmann relation is applied everywhere in the domain. For a plume simulation in a Geostationary Orbit (GEO) environment, the far-field charge density is low enough that the Debye length can exceed the cell size, while the Debye length is typically smaller than the cell size near the thruster exit. For this case, the electric potential calculation method is switched depending on a ratio of a Debye length to a cell size, rather than adjusting the grid resolution to apply one method everywhere in the computational domain.

1. Boltzmann Inversion

In the absence of magnetic fields and collisions with other species, the Boltzmann relation for an electron fluid can be derived by assuming spatially uniform velocity and that electrons react instantaneously to potential change. With these assumptions, only the electrostatic and pressure forces remain in the electron momentum equation so the two forces balance.

$$\nabla p = n_e e \nabla \phi \tag{6}$$

where p is the pressure, n_e is the electron density, e is the elementary charge, and ϕ is the electric potential. Potential values can be determined by solving Eq. (6) and assuming either a constant temperature or a polytropic relationship. When assuming a constant temperature with $p = nk_BT$, Eq. (6) reduces to

$$\phi - \phi^* = \frac{k_B T^*}{e} \ln\left(\frac{n_e}{n_e^*}\right) \tag{7}$$

where $*$ is the reference values and k_B is the Boltmann constant. On the other hand, the polytropic relationship is a more generalized thermodynamic process expressed as

$$\frac{p}{p^*} = \left(\frac{n}{n^*}\right)^{\gamma} = \left(\frac{T}{T^*}\right)^{\frac{\gamma}{\gamma-1}} \tag{8}$$

where γ is the polytropic index of any real number. Here, $\gamma = 1$ yields the isothermal (constant temperature) model and $\gamma = 5/3$ implies an adiabatic condition. The polytropic relationship yields an expression for electric potential when substituted in Eq. (6) and integrating the equation by parts.

$$\phi - \phi^* = \frac{k_B T^*}{e} \frac{\gamma}{\gamma - 1} \left\{ \left(\frac{n_e}{n_e^*}\right)^{\gamma-1} - 1 \right\} \tag{9}$$

When the Debye length is smaller than the grid resolution, the electron density is high enough to shield ion clouds such that the plasma is quasi-neutral within a cell. For this condition, ion density approximately equals electron density, and the ion density can be used in place of the electron density when solving for the potential values. The calculation of potential by the Boltzmann relation is originally implemented in AQUILA,[10,11] a module in COLISEUM framework, and has been imported to SM/MURF (`LogicalPotentialBoltzmannOp`). In order to use this operation, a user of the code needs to input three different reference parameters. These parameters reflect values at a point near a thruster exit that the Boltzmann relation holds and can be obtained experimentally as well as from a numerical code of a thruster.

2. Poisson Solve

When the resolution of SMesh is on the order of or smaller than the Debye length, the quasi-neutral assumption of plasma may no longer be valid. Therefore, the Boltzmann relation cannot be used to compute the potential values. For this case the Poisson's equation,

$$\nabla^2 \phi = -\frac{\rho_i - \rho_e}{\varepsilon_0}, \tag{10}$$

must be solved directly, where ε_0 is the permittivity of a free space, ρ is the charge density, and subscripts i and e are for ions and electrons, respectively. The ion charge density is obtained directly from the ion particles, but the electron charge density can only be approximated as electrons are not tracked in SM/MURF plume simulations. As a zeroth order approximation, a quasi-neutrality approximation can be enforced such that $\rho_e = \rho_i$ and the right hand side of Eq. (10) reduces to zero. If the electrons can be represented as a fluid in Boltzmann equilibrium, then the electron density can be determined from the Boltzmann relation.

$$\rho_e = \rho_e^* \exp\left(\frac{\phi - \phi^*}{T_e^*}\right) \tag{11}$$

where the reference parameters are the values at far-field and are not the same as in Eqs. (7) and (9). If these parameters cannot be obtained experimentally, their values need to be adjusted in a way that the simulation result is reasonable.

As of current, SM/MURF does not have any module to solve a linear system of equation. Alternatively, the Poisson's equation is solved by the element-wise Gauss-Seidel method, requiring an iteration until a residual of Eq. (10) meets a user-specified criterion. During the first time-step, the iterative solver may require tens of seconds to obtain a converged potential solution if an initial guess is not close to the solution. However, from a time-step to the next time-step, the potential solution does not change significantly so the simple iterative solver converges quickly when using the solution from the previous time-step as an initial guess.

3. Hybrid Method

As originally implemented in AQUILA,[11] SM/MURF has a flag to mark cells as non-neutral, quasi-neutral, and inside of a spacecraft component. If a cell is within a spacecraft component, the cell potential is set to the input component potential. For a cell in free space, the degree of quasi-neutrality is determined by comparing a Debye sphere volume ($V_{\text{Debye}} = 4/3\pi\lambda_D^3$) and the cell volume ($V_{\text{cell}}$).[12] If $\alpha V_{\text{Debye}} < V_{\text{cell}}$, the cell is flagged as quasi-neutral, and the potential is solved vie Boltzmann relation. Here, α is a factor to increase or decrease the region to be solved by one solver. The remaining cells that are flagged as non-neutral are solved by the Poisson solver with the other cells acting as Dirichlet boundary condition. Figure 6 shows two stages of computation of potentials. In Fig. 6, the left half represents the potential solution after the Boltzmann solver and the right half represents the potential solution after the values are filled by the Poisson solver. Note that the electron density is simply assumed to be the same as the ion density in this example. The Poisson solver clearly fills the remaining cells smoothly toward the domain boundaries with Neumann boundary condition.

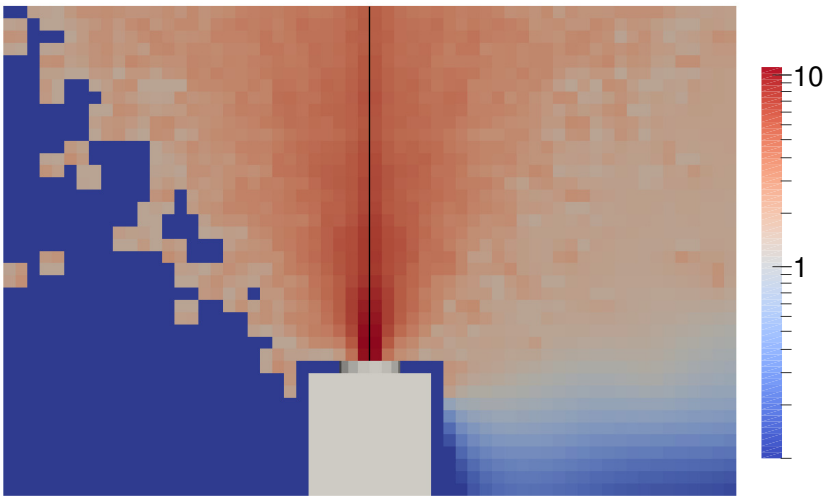


Figure 6. Electric potential solved only by Boltzmann inversion (left half) and by hybrid Boltzmann/Poisson solver (right half).

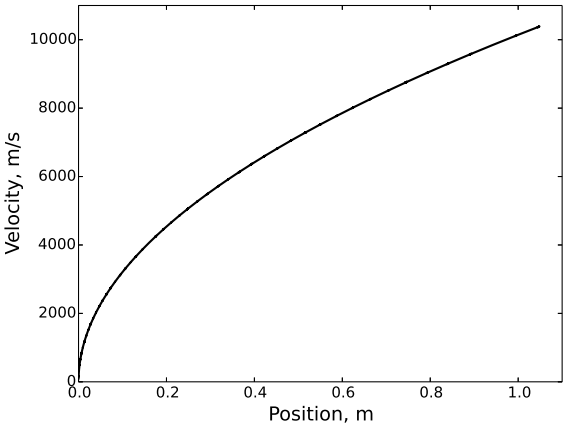


Figure 7. Velocity vs position of sampled particles during the integration test for the electrostatic push.

D. Electric Field Calculation and Electrostatic Particle Push

In PIC models, electric fields are computed by applying a finite difference equation to $E = -\nabla\phi$. Current version of SM/MURF computes node-centered electric fields using the electric potential values at the cell-centers. In the 3D structured mesh, eight cells share the same node. Therefore, the electric field at a node is determined by potentials of the eight surrounding cells. The electric field component in x-direction, E_x, at a cell represented by indices of i, j, and k is given by

$$E_{x,i,j,k} = -\frac{1}{4\Delta x}\sum_{j}^{2}\sum_{k}^{2}(\phi_{i,j,k} - \phi_{i-1,j,k}) \tag{12}$$

Here, a node at a lower corner of a cell has the same indices as the cell by the SM/MURF convention. The y- and z-components of the electric field are determined in the similar manner. The electric field calculation is such a simple operation that no unit test has been incorporated. The electric field calculation is incorporated in the integration test for a simplified plume simulation where results from SM/MURF and COLISEUM are compared.

The current version of SM/MURF uses the simplest time-integrator, namely an explicit Euler method. The Lorentz equation in an absence of magnetic field is expressed as $m\ddot{x} = qE$, where $\ddot{x}$ is the acceleration. Integrating the equation twice, a particle position after a single time-step, Δt, can be determined.

$$x_{m+1} = \frac{q}{2m}E\Delta t^2 + v_m\Delta t + x_m \tag{13}$$

where the subscript m denotes the current iteration step. The electric field is interpolated at x_m using the field values evaluated at eight corners of a cell.

In the integration test, xenon ions are injected to the domain at 100 m/s along the z-axis at every time-step. A constant electric field of 70 V/m in z-direction is applied throughout the domain. Depending on the number of time-steps that particles are advanced after they are introduced in the domain, the particles are at different velocities, which can be analytically determined. Once the simulation is completed, particles are sampled as shown in Fig. 7, and a polynomial fit to particles' velocity vs position curve is constructed. The coefficients of the curve-fit should be equivalent to the initial velocity and the applied electric field.

E. Particle-Surface Interaction

In the SM/MURF plume simulation, the `MOVE` stage in Fig. 3 involves a loop to ensure that all the particles are advanced by Δt. This loop is primarily used to bounce and sputter particles off of surfaces. Particles reaching $t = t_{m+1} = t_m + \Delta t$ are moved from `P-GST` to `P-DST` particle distributions, while particles impacting surfaces and newly added sputtered particles remain in `P-GST` distribution with particle time of $t_m < t < t_{m+1}$. For the next sub-step, particles in `P-GST` are advanced by $t_{m+1} - t$, and surface interaction calculation is

performed if any particle still hits a surface. These calculations are repeated until the number of particle in `P-GST` becomes zero.

Checking if particles impact surfaces can be very expensive, depending on the mesh resolution and the simulating geometry. A large number of surface elements means more surfaces per particles to scan through. On the other hand, a larger surface mesh area within a volume where particles reside means more frequent complete surface intersection check per particles. Therefore, the algorithm must minimize the number of particle-surface combinations for intersection check in order to be efficient. Current version of SM/MURF uses a list of triangular surface mesh elements per background mesh cell to determine possible combinations of particle and surface (`MSPDistSugarcubeSurfIntersectionOp`). A cell that a particle resides in is found from a particle position, and a list of triangles that intersect with the cell as well as the neighboring cells is determined during the "init" routine. Currently, the list only extends to one layer of neighboring cells such that particle-surface intersection check is performed only for triangles intersecting a block of 18 cells. This limits the particle's allowable Courant number to be three so particles can only move $3\Delta x$ at maximum, otherwise particles may never be aware of some triangular elements. If there is any surface mesh element in the list, the code determines all the intersections between every possible combination of the particles and listed triangular elements and then chooses one element that the particle intersects the soonest. The particle-surface intersection calculation is essentially the same as finding an intersection between a line segment and a triangle. The line segment is bounded by $x(t)$ and $x(t_{m+1})$. For the segment-triangle intersection test, we use the Moller-Trumbone method,[13] which is fast and most common algorithm. In addition to the standard Moller-Trumbone intersection check procedure, we have added the minimum criterion that checks if the two boxes bounding the segment and triangle intersect. In this way, the heart of intersection check calculation can be skipped for many of the particle-triangle combinations, especially for the case where the surface mesh resolution is coarser than the background mesh resolution. When the algorithm determines that the particle intersects a surface, the operation checks the direction of intersection using the particle velocity and the surface's normal vector. This is to prevent particles to be stuck within a geometry in case of particles leaking through the surface mesh unexpectedly.

The particles intersecting with triangular elements then either stick or reflect at the intersection point, depending on the user-defined interaction for each material combination (`MSPDistSurfaceInteractionOp`). The sticking coefficient, C_{stick}, determines the probability that the particle sticks to a surface. If $U < C_{\text{stick}}$ where U is a random number between 0 and 1, then the particle is killed and is no longer tracked. Otherwise, the particle is reflected off the surface. Current version uses three independent parameters to smoothly mix between specular and diffuse reflections. These parameters are coefficients of diffuse reflection (C_{diff}), restitution (C_{rest}), and thermal accommodation (C_{accom}). When a surface is perfectly smooth, particles can experience a specular reflection and be reflected in a mirror-like manner. The velocity tangent to the surface remains the same, but the normal velocity becomes in the same direction as the surface normal.

$$\vec{v}_{\text{spec}} = \vec{v} - 2\vec{v}_n \tag{14}$$

where $\vec{v}_{\text{spec}}$ is the velocity vector after reflection, $\vec{v}$ is the velocity vector before reflection, and $\vec{v}_n$ is the normal component of v. The diffuse reflection occurs when particles are reflected on Lambertian surfaces. The distribution follows the cosine law, and the polar angle probability distribution function is expressed as $g(\theta) = \sin 2\theta$.[14] The velocity sampling for diffuse reflection is done by the Box-Muller method.[7]

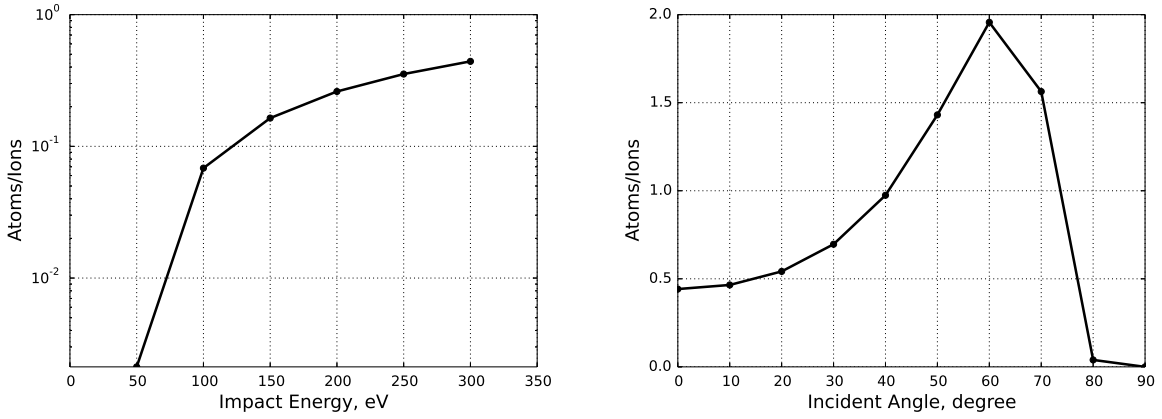
$$v_t = v_{th}\sqrt{-\log(U_1)}, \quad v_n = v_{th}\sqrt{-\log(U_2)} \tag{15}$$

where U_1 and U_2 are random numbers, and subscripts t, n, th represent tangent, normal, and thermal, respectively. Here, the thermal velocity is given by $v_{th} = \sqrt{2k_bT_s/m}$, and T_s is the surface temperature. The diffuse reflection velocity vector, $\vec{v}_{\text{diff}}$, is obtained by transforming the surface-aligned coordinate system to the Cartesian coordinate system. The coefficient of diffuse reflection mixes the directions from the specular and diffuse reflections.

$$\hat{v} = \hat{v}_{\text{spec}} + C_{\text{diff}}(\hat{v}_{\text{diff}} - \hat{v}_{\text{spec}}) \tag{16}$$

where $\hat{v}$ is the unit vector of $\vec{v}$. The restitution coefficient is the ratio of the post-impact to the pre-impact speed for specular reflection, where as the accommodation coefficient is related to the energy transfer between a particle and a surface. The magnitude of velocity after reflection then becomes as follows.

$$v'_{\text{spec}} = C_{\text{rest}}v_{\text{spec}}, \qquad v = v'_{\text{spec}} + C_{\text{accom}}(v_{\text{diff}} - v'_{\text{spec}}) \tag{17}$$



(a) (b)

Figure 8. Total sputtering yield curves for xenon ion impact on aluminum as functions of (a) energy at normal incidence and (b) angle at an energy of 300 eV.

When ions impact the surfaces, they can recombine with electrons. Therefore, changing the species during the surface reflection operation is also incorporated. No unit test has been incorporated for surface reflection calculations. However, these surface reflection calculations are performed in several integration tests for other operations.

For each combination of incident and target materials, the sputtered material and the sputtering model with appropriate coefficients can be specified in the input file. Currently, seven different models for total sputter yield are implemented: the constant yield model and models by Matsunami et al.,[15] Yamamura and Tawara,[16] Kannenberg et al.,[17] Roussel et al.,[18] Garnier et al.,[19] and Pencil et al.[20] Yamamura and Tawara's model is the most general and agrees fairly well with many combination of source and target materials. Examples of sputtering yield curves as functions of energy and angle are shown in Fig. 8. Figure 8 was produced for xenon ion impact on aluminum surface. References 17–20 focus on xenon ions sputtering satellite and Hall thruster materials so the applicable source/target material combination may be limited. The angular distribution of sputtered particles can be determined by either a cosine distribution or the distribution slightly modified from Zhang's model[21] which was based on the work by Yamamura.[22,23] The outputs from the models are compared against the experimental data to confirm the correctness of the implementation, and then these comparisons are incorporated as unit tests.

F. Numerical Probes

SM/MURF currently has operations to collect current densities and measure energy distributions that mimic Faraday and Retarded Potential Analyzer (RPA) probes, respectively. These numerical probes can be attached and fixed to a component of a surface mesh (`MSPDistProbeFixedOp`) or virtually scan along a spherical surface at discrete polar angles (`MSPDistStageSphericalOp`). Fixed probes are useful when a probe is meshed or an area-averaged value of current density needs to be monitored. Otherwise, a more detailed current density profile can be obtained from surface mesh fields. Note that a surface mesh component is set to "virtual" such that particles do not see the fixed probe surface. Spherical stage probes are useful when constructing an angular distribution of charged particle source. Since this operation does not really require a surface mesh, extracting the same information from the surface mesh field requires a bit of effort, e.g. one needs to generate a spherical mesh to collect charged particles. When collecting charged particles for a RPA probe on a spherical stage, they are distributed to a two-dimensional array as functions of polar angle and incident energy. The first-order weights are used to distribute currents to the two nearest energy bins. Summing up the charged particles at the same polar angle, the results reduces to the current collected by a Faraday probe.

Integration tests for both the fixed and spherical stage probes have been incorporated. In the test for a fixed probe, particles of two species are injected at constant velocities from one end of a cube and are collected at the other end where the probe is attached. The test script checks if the species currents and

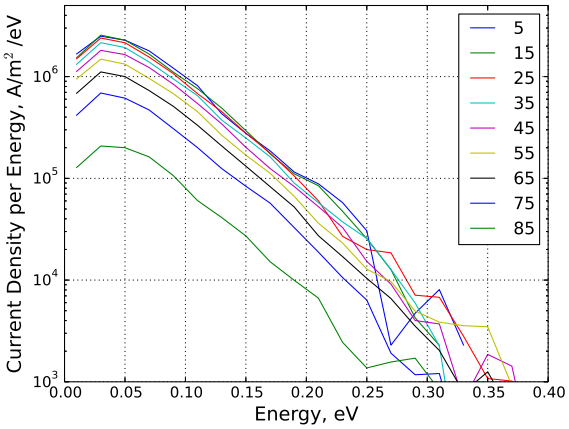


Figure 9. RPA data from a simulation for an integration test. The numbers in the legend are the polar angles.

velocities are consistent with the input parameters. The script also checks if the total currents of Faraday and RPA probes are the same and if the total current is equivalent to the sum of species currents. The test for a spherical stage probe is similar to the fixed probe test. The only difference is the set-up where particles of two species are injected from a disk much smaller than the spherical stage with temperature values of 400 K and 300 K but with a normal velocity of 0 m/s. Figure 9 shows the numerical RPA data obtained from this integration test. The current density per energy is summed over species. The current density per energy is peaked at an energy corresponding to a temperature between 300 and 400 K.

G. Particle Sampling for SMesh Field Calculation

The zeroth order method for charge deposition is prone to a noisy charge density distribution, occasionally making a PIC simulation inevitably unstable. In contrast, a high order method smooths out the distribution such that important feature of plasma may be masked. For this reason, the first order method is commonly used in a PIC simulation. The high order weighting method can be important for unsteady problems, where mixing cannot be performed between different time-steps. A similar result can be achieved by applying a low noise filter with multiple passes.[24] Unlike the PIC models, the DSMC codes typically use the zeroth order method and mixes between time-steps. Although this method will retain the flow feature within the size of a cell, the mixing needs to be started after the solution reaches steady-state, and it takes a good number of time-steps because the convergence only scales with a square root of the number of sampled particles.

Besides the operation for charge deposition, SM/MURF has an operation to sample particles, velocities, and velocity squares in a DSMC-like way to compute field data of density, velocity, and temperature (`MSPDistSampleOp`).

$$P_i = \sum_p^{N_i} w_p, \quad V_{j,i} = \sum_p^{N_i} w_p v_{j,p}, \quad W_{j,i} = \sum_p^{N_i} w_p v_{j,p}^2, \tag{18}$$

where w is particle's specific weight representing the ratio of real particles to simulation particles and the subscripts p, i, and j are the indices for particles, cells, and directions, respectively. When mixing is turned on, these sampled quantities are mixed with the same quantities averaged through m number of iterations.

$$P_{i,m} = \frac{1}{m}\left[P_i + (m-1)P_{i,m-1}\right] \tag{19}$$

The other sampled quantities are mixed in the similar manner. The sampling method described herein assumes a constant time-step and will be upgraded to use time instead of iteration count to be compatible with variable time-steps. Note that the sampling method is already capable of taking variable specific weights.

The field data are computed based on the sampled quantities only before they are written to a file for

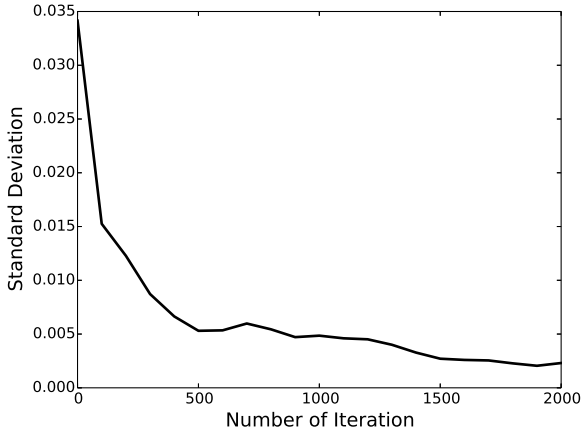


Figure 10. Convergence of particle density with the number of sampling.

visualization. For each species, the field data can be calculated by[6]

$$n = P_{i,m}/V_{\text{cell}}, \quad v_j = V_{j,i,m}/P_{i,m}, \quad T_j = \frac{1}{R}\left(\frac{W_{j,i,m}}{P_{i,m}} - v_j^2\right) \tag{20}$$

where R is the specific gas constant. The total velocity and temperature are given by $v = (v_x^2 + v_y^2 + v_z^2)^{0.5}$ and $T = (T_x + T_y + T_z)/3$, respectively. When computing the overall velocity and temperature field data summed over species, the field data in Eq. (20) are weighted by the species' specific gas constant.

In the integration test for SMesh sampling, particles of two different species are injected to a cubic domain at Maxwellian distributions at different temperatures and streaming velocities. The SMesh field data from the code at the end of simulation should correspond to the input parameters. The tested flow field data are the density, speed, streaming velocity, and temperature for the two species as well as total. The sampling calculation starts when steady-state condition is established, and the flow properties should be uniform at the time. The integration test checks if the maximum deviation of the flow field from the input value is within 99.9999 % confidence interval. Figure 10 shows the convergence of particle density with the number of sampling. The standard deviation in the plot is equivalent to the L2 norm of the statistical error. As expected, the statistical noise generally decreases with the number of iteration.

H. Elastic Collision

In the plumes of electric propulsion devices, ions that are electrostatically accelerated to high energy can experience elastic collisions with slow neutral atoms exiting the thrusters. During the collisional process, the high energy ions exchange momentum with their collision partners, and some fraction of the population is deflected at angles greater than the plume divergence. Meanwhile, the elastic collision may involve an exchange of one or more electrons, leading to a slow CEX ion and a high energy atom after each charge-exchange (CEX) process. Therefore, the CEX collision is considered as a subset of elastic collision, and we call the elastic collision without an exchange of charge a MEX collision. Proper modeling of these collisions is particularly important in accurately determining the plume current profile and assessing the integration of spacecraft components as well as the thruster life and long duration performance; the slow CEX ions are prone to electric fields that direct the ions toward the spacecraft and thruster components, resulting in contamination through sputtering and deposition after gaining significant energies.

The elastic collision calculation in our plume simulation is typically performed with the Monte Carlo Collision (MCC) method, in which particles are collided with a background fluid. The MCC method does not conserve momentum, unlike the DSMC method which performs detailed collisions between particles to nearby particles. In the MCC routine, a probability that a particle experiences a MEX or CEX collision is expressed as

$$P_{\text{coll}} = 1 - \exp(-\Delta t \sigma v_r n_o) \tag{21}$$

where σ is the collision cross-section, v_r is the relative velocity between ions and neutral atoms, and n_o is the background neutral density. This equation gives a positive value that is always less than or equal to 1. Whether a collision event takes place is determined by comparing P_{coll} with a random number, U, uniformly probable between 0 and 1. If $P_{\text{coll}} > U$, then the particle experiences an elastic collision.

The CEX collision cross-section has been measured experimentally and fitted to a simple logarithmic formula by Miller et al.[25]

$$\sigma_{\text{CEX}} = 87.3 - 13.6 \log(E) \tag{22}$$

In many models, the MEX collision cross-section, σ_{MEX}, is typically assumed to be the same as σ_{CEX}. This is not a bad assumption as the MEX and CEX collisions are nearly the same collisional process except for the exchange of charge.

The current version of elastic collision operation (`MSPDistMCCElasticFitOp`) treats the MEX and CEX collisions as independent collision mechanisms such that one of the two collisions can be turned off. The operation is based on the model implemented in COLISEUM, which uses curve-fit representations of laboratory frame (LAB) differential cross-sections to sample the scattering angles.[26]

$$\left.\frac{d\sigma}{d\Omega}\right|_{\text{LAB}} = \theta^{A_{\text{MEX}}} 10^{B_{\text{MEX}}} + (90-\theta)^{A_{\text{CEX}}} 10^{B_{\text{CEX}}} \tag{23}$$

where θ is the angle in degrees, and A and B for MEX and CEX collisions are fitting coefficients to the differential cross-section at an ion energy per unit charge of 300 V given in table 2 . Integrating the differential cross-sections for all angles, the values should correspond independently to the total cross-sections for MEX and CEX collisions. In doing this, some assumptions at the extreme angles near 0° and 90° must be imposed. The model assumes that the differential cross-sections are constant past some critical angles, θ_{cr}, and the values for θ_{cr} are given in table 2.

The LAB differential cross-sections for the MEX and CEX collisions are transformed in terms of the center of mass (CM) scattering angle, χ. This process is necessary in a proper collision calculation that includes the effect of the target particle velocity. Then, the cumulative distribution function, together with the acceptance-rejection routine, can be used to sample the MEX and CEX collision angles.

$$\frac{\int_0^{\chi} \frac{d\sigma}{d\chi'} d\chi'}{\int_0^{\pi} \frac{d\sigma}{d\chi'} d\chi'} = \frac{\int_0^{\chi} f(\alpha) d\alpha}{\int_0^{\pi} f(\alpha) d\alpha} = U \tag{24}$$

where the comparison function, f, can be expressed as

$$f(\chi) = \begin{cases} \theta^A & \text{for } \chi < \theta_{\text{cr}} \\ \chi^A & \text{for } \chi \geq \theta_{\text{cr}} \end{cases} \tag{25}$$

For every particle that is determined to experience the specific collision process, the calculation of post-collision velocity involves multiple coordinate transformations. The calculation procedure can be found elsewhere.[8,26,28]

The integration test for the collision operation is performed by injecting ions at 300 eV (`MSPDistNormalMaxwellianStreamOp`), applying collisions (`MSPDistMCCElasticFitOp`), and writing the particle distribution to a file (`MSPDistWriteVTKOp`). From the particle data, the LAB differential cross-section can be constructed and compared with Eq. (23). The 2-norm and maximum-norm are checked to ensure they are within some pre-defined bounds. Figure 11 shows the differential cross-sections for the MEX and CEX collisions. The curve-fits for the MEX and CEX collisions are obtained from the first and second terms of right hand side expression in Eq. (23).

Table 2. Fitting coefficients for differential cross-sections at an ion energy per unit charge of 300 V.[26,27]

Species	Mechanism	A	B	θ_{cr}
$Xe^+ + Xe$	MEX	−2.02	3.24	3.53×10^{-5}
	CEX	−1.098	−1.53	1.37×10^{-3}
$Xe^{2+} + Xe$	MEX	−2.34	3.17	0.059
	CEX	−1.43	1.47	0.143

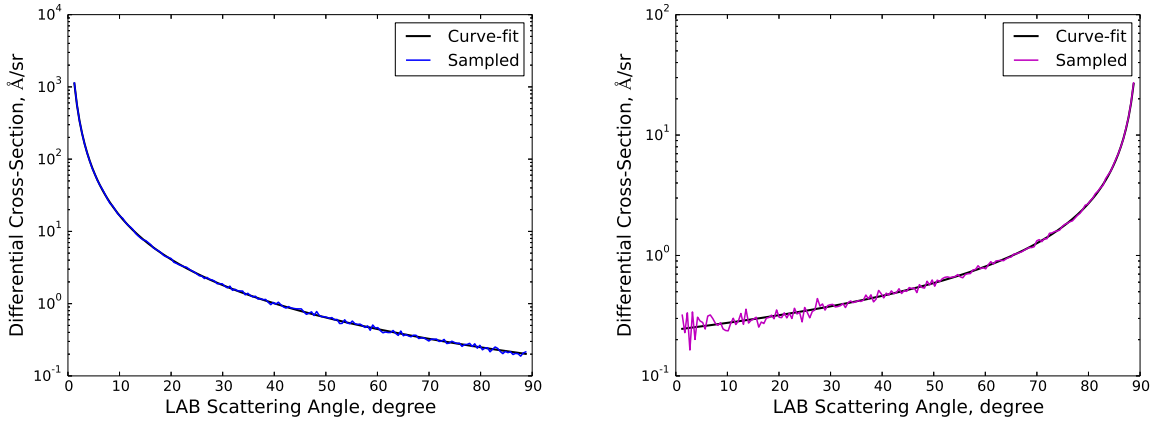


Figure 11. Laboratory frame differential cross-Sections for (a) MEX and (b) CEX collisions at an ion energy of 300 V.

I. Surface Mesh Field Calculation

Similarly to the SMesh field calculation described in Sec. IV.G, the number of particles, normal and tangent components of momentum, and the kinetic energy are sampled on the unstructured surface mesh, UMesh.

$$M_{n,i} = \sum_p^{N_i} m_p w_p v_{n,p}, \quad M_{t,i} = \sum_p^{N_i} m_p w_p v_{t,p}, \quad KE_i = \frac{1}{2}\sum_p^{N_i} m_p w_p (v_{x,p}^2 + v_{y,p}^2 + v_{z,p}^2) \tag{26}$$

Here, the subscript i, n, and t represent the surface element index, normal component, and tangent component, respectively. The number of particles to an element, P_i, is given by the same expression as in Eq. (18). The sampling quantities on UMesh is mixed between iterations just as in Eq. (19). Denoting the averaged quantities with an additional index m, the UMesh field data are computed as follows.

$$\dot{N} = \frac{1}{S_{\text{el}}\Delta t_s} P_{i,m}, \quad p = \frac{1}{S_{\text{el}}\Delta t_s} M_{n,i,m}, \quad \tau = \frac{1}{S_{\text{el}}\Delta t_s} M_{t,i,m}, \quad q = \frac{1}{S_{\text{el}}\Delta t_s} KE_{i,m} \tag{27}$$

where S_{el} is the area of a mesh element, Δt_s is the sampled time-step, $\dot{N}$ is the particle flux, p is the pressure, τ is the shear stress, and q is the heat flux.

The integration test for the UMesh field calculation is very similar to the one for the SMesh field. Particles of two different species are injected to a cubic domain at Maxwellian distributions at different temperatures and streaming velocities. The UMesh field data are computed at the opposite side from the particle injection plane as shown in Fig. 12. The UMesh field data from the code at the end of simulation should correspond to the field data analytically evaluated with the input parameters. The analytical expressions for $\dot{N}$, p, and q are given by the following equations.[6]

$$\begin{aligned}
\dot{N} &= \frac{\dot{m}}{mS} \\
n &= \frac{2\sqrt{\pi}\beta\dot{N}}{\exp(-s_n^2) + \sqrt{\pi}s_n[1+\text{erf}(s_n)]} \\
p &= \frac{mn\left(s_n\exp(-s_n^2) + \sqrt{\pi}[1+\text{erf}(s_n)](0.5+s_n^2)\right)}{2\sqrt{\pi}\beta^2} \\
q &= \frac{mn\left((s_n^2+2)\exp(-s_n^2) + \sqrt{\pi}[1+\text{erf}(s_n)](2.5+s_n^2)\right)}{4\sqrt{\pi}\beta^3}
\end{aligned} \tag{28}$$

where S is the total area of particle injection. The sampling calculation starts when a steady-state condition is established, and the UMesh field data should be nearly uniform across the surface elements as shown in Fig. 12. While a large number of simulation particles is preferable in obtaining a distribution with a low

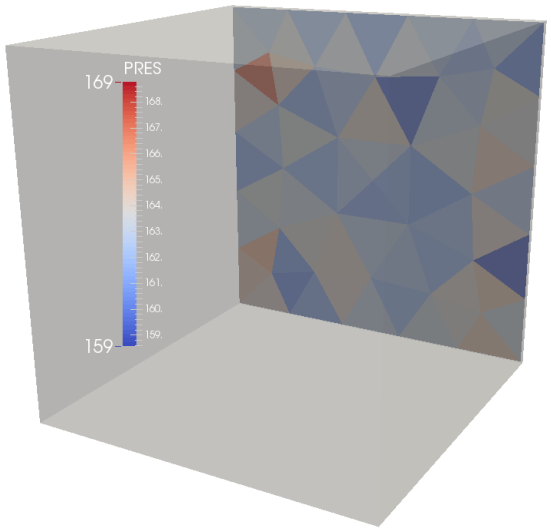


Figure 12. Pressure field on downstream surface with a cube representing the simulation domain.

statistical noise, it leads to longer simulation time, but the integration test needs to be fast enough such that the regression test can be finished within a reasonable amount of time. For this reason, the number of simulation particles are kept to be small, but the bounds to accept the outputs from the integration test are set to be reasonably large.

J. Particle Patch across Multiple Sub-Domains

In the SM/MURF simulation, particles located within the ghost cell region and outside the domain are deleted from the particle distribution. This is done by first marking these particles by setting their cell IDs to be greater than the number of SMesh cells (`MSPDistCellIDOp`), copying these particles to another particle distribution (`MSPDistSplitOp`), and removing these particles from the original distribution (`MSPDistRemovePartOp` or `MSPDistSortOp`). For a simulation with multiple sub-domains, the program needs to patch particles across the sub-domains, otherwise particles are deleted before being transfered to a nearby sub-domain. The particle patching is performed with two additional operations, `GSOPatchOp` and `MSPDistCombineOp`. `GSOPatchOp` simply flags which GSObject to perform patching during the "init" routine. For this case, patching is performed on `MSPDist` that contains particles outside the interior sub-domain.[a] Then, at the end of a stage, these particles are copied to a particle distribution owned by another sub-domain as illustrated in Fig. 13. Finally, these particles are added to the main particle distribution by `MSPDistCombineOp`. A simple flowchart for particle patching is shown in Fig. 14.

The integration test for particle patching compares the particle positions and velocities at the end of simulation for single and dual domain cases. As long as the particles' initial conditions are set exactly in the same manner, both the single and dual domain cases should have identical particle outputs. In this simulation, particles are laid down within one half of the entire cubic domain (`MSPDistBoxICOp`). This is to ensure that particles are only created in the first domain such that both single and dual domain cases use the same sequence of random numbers in determining the particle positions and velocities. Then, particles are advanced by a fixed number of time-steps. At the domain boundaries, particles are reflected specularly. Figure 15 shows the particles at the end of the integration test simulation. It is clearly seen that particles from the single and dual domain runs are identical.

[a]Each sub-domain contains an interior region surrounded by layers of ghost cells. The interior sub-domain is referring to the interior region without the ghost layer

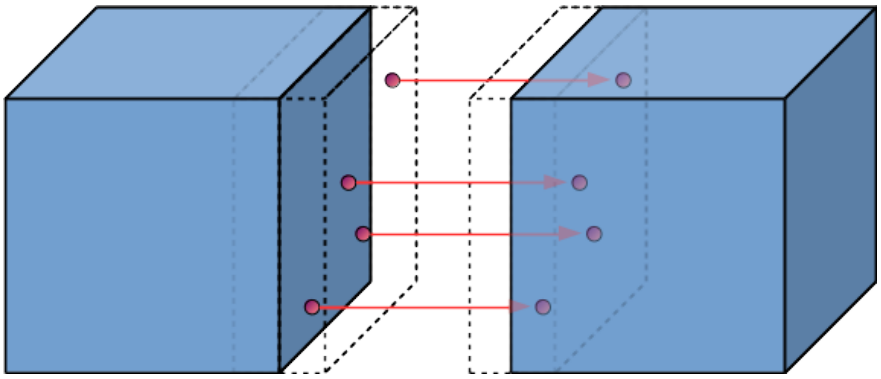


Figure 13. Illustration of particle patching. The blue cubes represent sub-domains, and dashed boxes represent the ghost layer. Particles located inside the ghost layer are copied to the other sub-domain. The two domains are drawn apart from each other for illustration purpose.

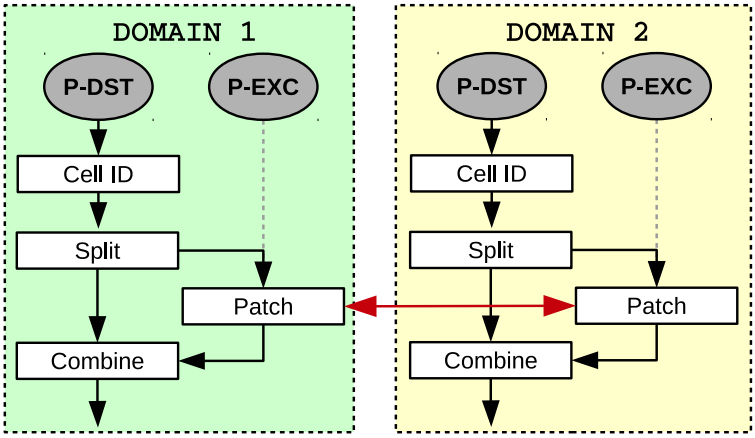
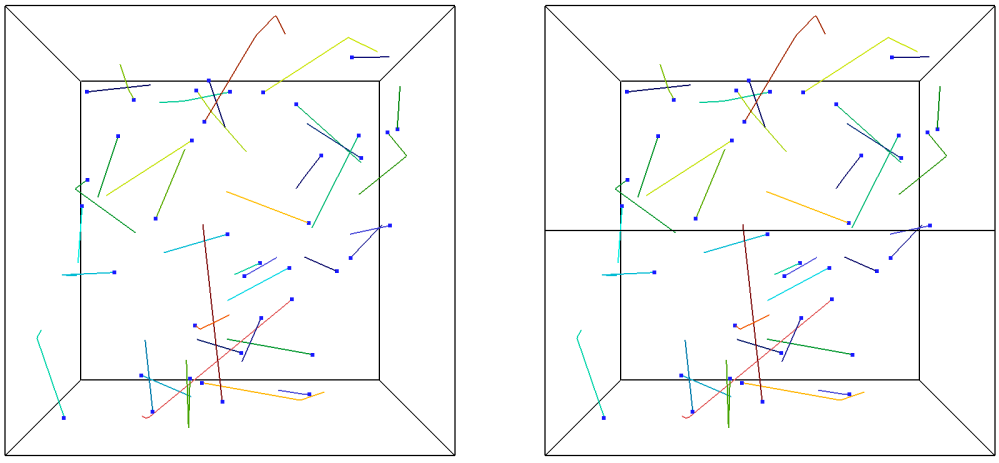
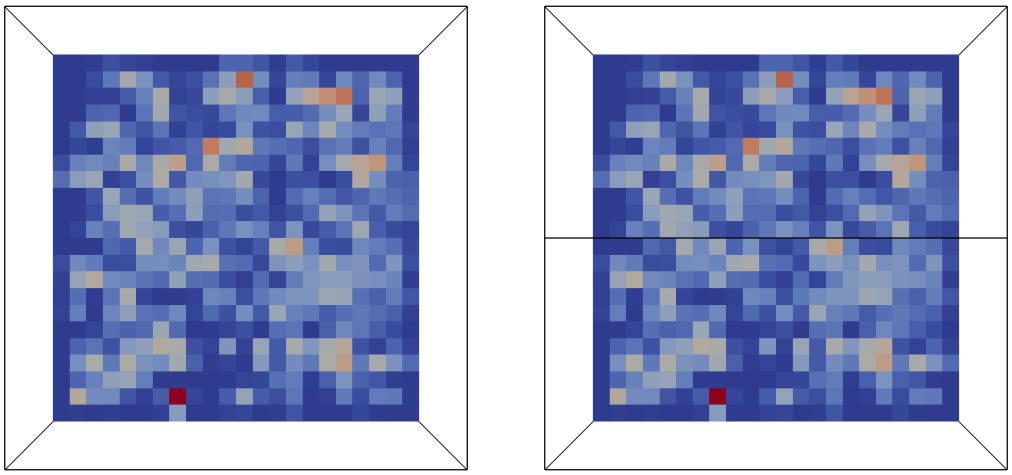


Figure 14. Flowchart for particle patching. `P-DST` and `P-EXC` are the `MSPDist` object class that holds particle information, and each sub-domain possesses the two particle distributions. Particles are exchanged across sub-domains at the red arrow.



(a) (b)

Figure 15. Particles after 2,000 time-steps for (a) single and (b) dual domain simulations.



(a) (b)

Figure 16. Charge density at the end of simulation for (a) single and (b) dual domain simulations.

K. Field Patch across Multiple Sub-Domains

Once it is verified that particles patching is properly performed, as described in Sec. IV.J, the patching of SMesh field data can be tested. For plume simulations, the only field data that are patched across the sub-domains are the ones that use a first order interpolation such as the charge density. Field data that use a zero-th order method, as described in Sec. IV.G, are not patched. When using the first order method for charge deposition, the particle charges can be distributed to the first layer of ghost cells. These charges need to be added to corresponding cells that are not ghost cells. Similarly to `GSOPatchOp`, the field to be patched is set in an operation, `LogicalFieldPatchOp`. Then, the field data are patched at the end of stage.

The integration test for field patching is very similar to the one for particle patching. Ions are created within one half of the entire domain such that they are created only within the first domain for both the single and duel sub-domain simulations. Ions are moved within the cubic domain, and specular reflection is applied when they cross the domain boundaries. To minimize the sources of error, the ion motion does not account for the electric field; this decouples the charge density calculation with the particle motion. For the duel sub-domain case, particles are patched across the two sub-domains. At the end of simulation, the charge density fields are compared between the single and dual sub-domain cases. If the field patching is not performed properly, the charge density field becomes inaccurate near the sub-domain boundaries. Figure 16 shows the charge density from the two simulations, and the field data are identical.

L. Particle Restart

At every user-defined number of time-steps, the particle information can be written to a file in a VTK format (`MSPDistWriteVTKOp`). Every file can be stored to visualize the particle trajectories, or one file can be kept overwritten to only store a restart file in order to save space on a hard drive. In a simulation with multiple sub-domains, particles within each sub-domain are written to a file such that the number of files per time-step corresponds to the number of sub-domains. However, the particle data are really independent of how the domain is partitioned and the grid resolution. In other words, the number of particle data files per time-step is not required to coincide with the number of sub-domains when particle data are to be read during a restart of a simulation. Each sub-domain opens all the particle data files from the same time-step and only gathers particles that are within the interior region of the sub-domain (`MSPDistReadVTKOp`). This feature does not exist in COLISEUM and adds much flexibility by enabling changes to the number of sub-domains and grid resolution between restarts. The particle data files contain the world time and iteration number from the previous run, and these parameters can be updated in the new simulation to be started.

The integration test for particle restart is performed again on a simulation with a domain where particles filled only on the half of the domain. The simulation is first started with two sub-domains and then restarted with one, two, and eight sub-domains. The particle positions and velocities should be exactly the same between all cases as well as a simulation run with only one sub-domain without a restart.

M. Field Restart

Unlike the case with particle data, the sampling for SMesh can only be restarted if the domain is partitioned in the exactly the same way as the simulation before the restart. This will need to be changed to support variable partitioning for dynamic load balancing. However, dealing with a change of SMesh resolution can be challenging. The restart for the field data is not available in COLISEUM, so this is a new feature added to SM/MURF. The restart file is overwritten at every user-defined number of time-steps (`SampleFieldWriteVTKOp`). The restart file includes the sampled quantities, $P_{i,m}$, $V_{j,i,m}$, and $W_{j,i,m}$, given in Eq. (18) for each species, the world time, the number of iterations, and the number of sampling steps. When restarting a simulation with the sampled quantities from the previous run (`SampleFieldReadVTKOp`), the world time and iteration number can be updated according to the number stored in the restart file. Since the sub-domains need to be set up in the same way as the previous run, reading the sampled quantities is trivial; each sub-domain reads data from a restart file corresponding to the sub-domain.

The set-up for the integration test simulation for field restart is nearly the same as the tests for particle patching, field patching, and particle restart. The test can be performed with any number of partitioning along x-, y-, and z-axes as long as it is consistent between restarts. The field data computed from the sampled quantities are compared with the field data obtained from a simulation with one sub-domain and no restart.

V. Verification of Result with COLISEUM

Up to this point, all of the individual pieces of SM/MURF relevant to plume simulations have been verified through unit and integration tests. Now, the final integration test compares results from baseline plume simulations of SM/MURF and COLISEUM. For this test, a simple geometry shown in Fig. 17 is chosen to make the comparison easier. The geometry consists of four different components: Thruster, Solar Array, Spacecraft Body, and Plate. These components are made of aluminum except for Plate which is made of iron. This test is also a part of regression tests; therefore, the simulation is required to be completed within a reasonable amount of time, i.e. $\lesssim 3$ hours. For this reason, the resolution of the surface mesh and Cartesian mesh is kept relatively coarse (see Fig. 17). Furthermore, the simulation is intended to be simpler than the actual full plume simulation in order to quickly identify the source of error.

Hall thruster source is attached to Thruster in a direction normal to the Thruster face, and three different gas species, xenon atom and singly and doubly charged xenon ions are injected as macro-particles from the source. Only the elastic collision between singly charged ions and neutral atoms is turned on in this simulation. The MEX ions contribute to the beam current larger than the plume divergence, and the CEX ions created from the collisional process can be directed toward the spacecraft components by the electric field. All of the surface reflection mechanisms are turned off such that incident particles can only stick to the surfaces. However, sputtering of iron surface by singly charged ions is turned on, so the sputtered iron atoms are introduced at the Plate surface and fly toward the spacecraft. These iron atoms can accumulate onto Thruster, Spacecraft Body, and Solar Array. Since the sputtering is dependent on the spatial and energy distribution of ions at the Plate surface, every core operation can affect the distribution of iron onto the spacecraft components, and errors from different operations can cascade through other operations. Mixing of field data is started once the simulation reaches steady-state, and the averaged field data from SM/MURF and COLISEUM are compared. The inherent difference between the two frameworks is due to the location that the field is computed on. More specifically, SM/MURF computes data on the cell-centers, and COLISEUM computes data on the nodes. This can affect the SMesh field and electric field, which then affects the ion trajectories. However, the results from the two codes should converge as refining the grid resolution.

Figure 18 shows the SMesh field data from the baseline plume simulation. The field data computed by `MSPDistSampleOp` are time-averaged over $20,000$ time-steps. The charge density shown in Fig. 18(e) is time-averaged with an operation `LogicalFieldTimeAverageOp`, but the instantaneous value is actually used in the electric potential calculation shown in Fig. 18(f). The HPHall source model provides particles of different species at different distributions. The xenon atoms are nearly at a cosine distribution such that the atoms are spread out into the free space. They are much slower than the other ion species, so the steady-state can only be reached once their distribution reaches steady-state. On the other hand, ions are generally very fast at the thruster exit due to the electrostatic acceleration within the thruster and are more focused in the normal direction. Due to the collisions applied to the singly charged ions, some ions are deflected at

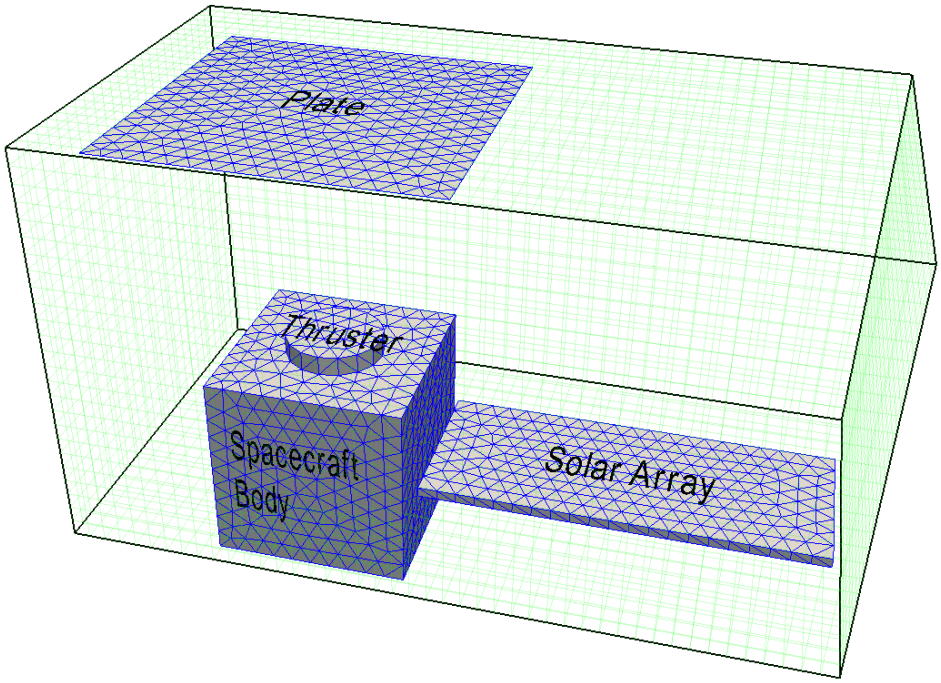
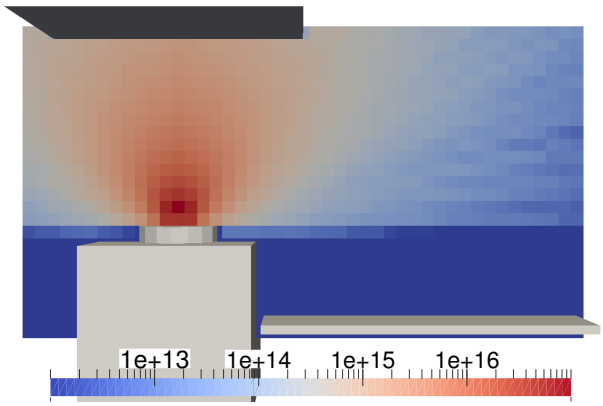


Figure 17. Geometry and meshes used in a plume simulation for an integration test.

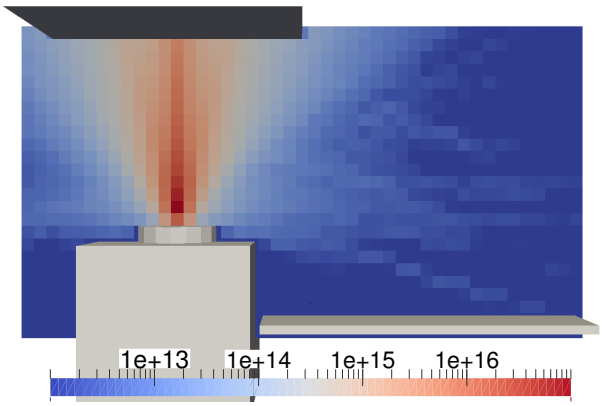
very large angles and can impact the spacecraft components directly as shown in Fig. 19(b). Even without collisions, small fluxes of neutral atoms (Fig. 19(a)) and doubly charged ions (Fig. 19(c)) are computed at the spacecraft surfaces, indicating that the HPHall source model predicts backscattering atoms and slow doubly charged ions. As the simulation is set-up, iron atoms are sputtered off from the downstream surface (Fig. 18(d)), and these atoms reach the spacecraft surfaces at some distribution (Fig. 19(d)).

Figure 20 shows the differences in SMesh field data between SM/MURF and COLISEUM. Note that the colormap range is kept the same as in Fig. 18 to clearly show the differences. In order to make a fair comparison between the two codes, the cell-centered data are approximated by taking an average of data at eight adjacent nodes. For the three species injected from the source, the differences in cells along the Thruster center are relatively large. The maximum norm of relative difference is approximately 65 % for most of species. This is mainly due to the averaging to obtain the cell-centered value and the Thruster center being aligned with the cell-centers. If the maximum in a distribution lies along a cell-center, this value can never be recovered from the nearby nodes where the values are always smaller than the maximum. Relatively large differences are also seen at the cells next to the Plate surface. Again, this is attributed to the cell-centered data. Half of the node volume is behind Plate such that the densities at these nodes are half, thus affecting the averaged field data at the corresponding cell-center location.

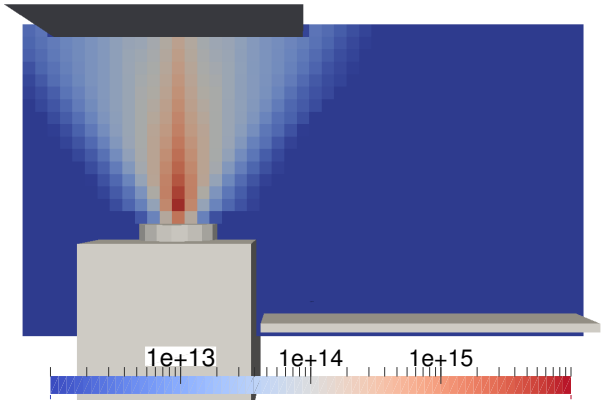
The difference in where the field data are computed affects the electric field calculation. Along the Thruster center axis, SM/MURF predicts a higher electric potential, which then results in a higher electric field, causing the charged particles to be accelerated more. Figures 21 and 22 show numerical probe data computed by SM/MURF and COLISEUM. For the spherical stage probes, data are obtained along a hemispherical surface whose radius is about a third of distance from the Thruster face to Plate and whose center coincides with the center of Thruster face. The numerical measurements are performed on discrete segments of the hemisphere, each covering 10° of polar angles, i.e. $0^\circ \leq \theta < 10^\circ$, $10^\circ \leq \theta < 20^\circ$, ..., $80^\circ \leq \theta < 90^\circ$. Since the current density increases at smaller angles as shown in Fig. 21(a), the RPA data with larger integrated area correspond to the segments with smaller polar angles. From the RPA data shown in Figs. 21(b) and 22, it can be seen that the SM/MURF data are shifted slightly to the right compared to the COLISEUM data, indicating that charged particles have gained more energy from the electrostatic force. Nevertheless, the agreement of faraday and RPA data between the two codes is excellent.



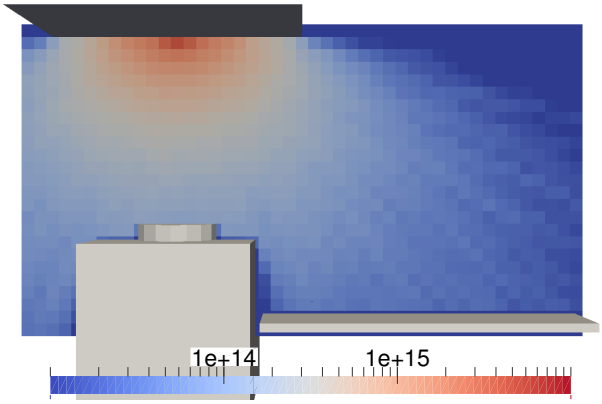
(a) Xenon atom density



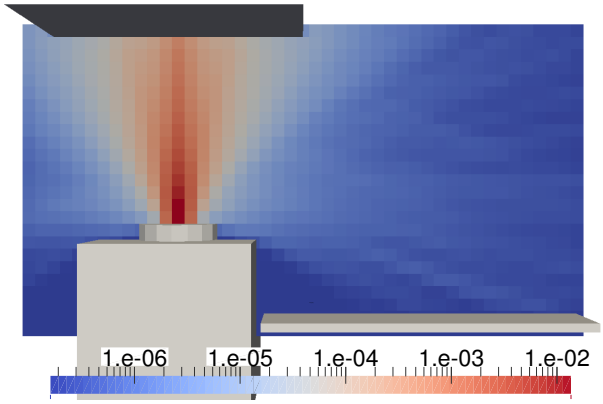
(b) Singly charged ion density



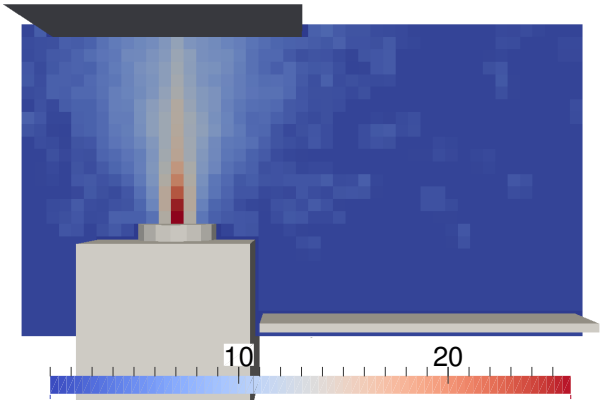
(c) Doubly charged ion density



(d) Iron atom density

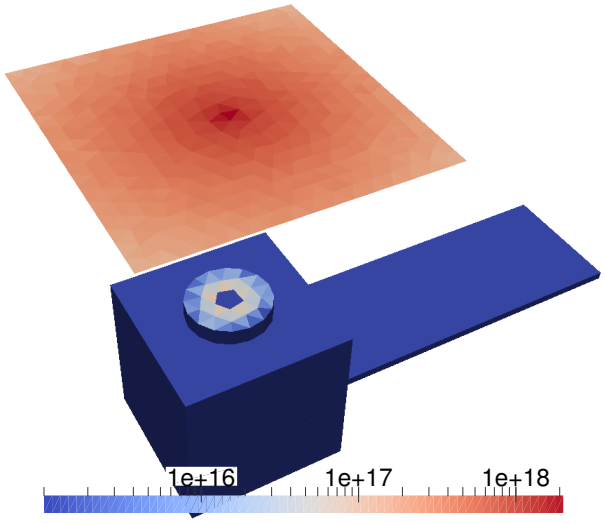


(e) Charge density

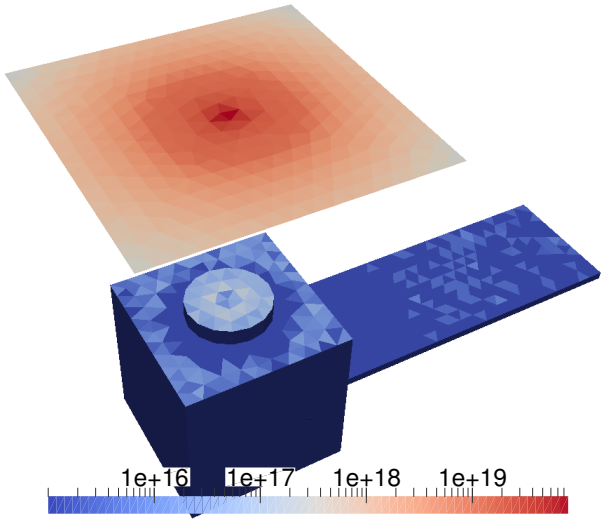


(f) Electric potential (instantaneous)

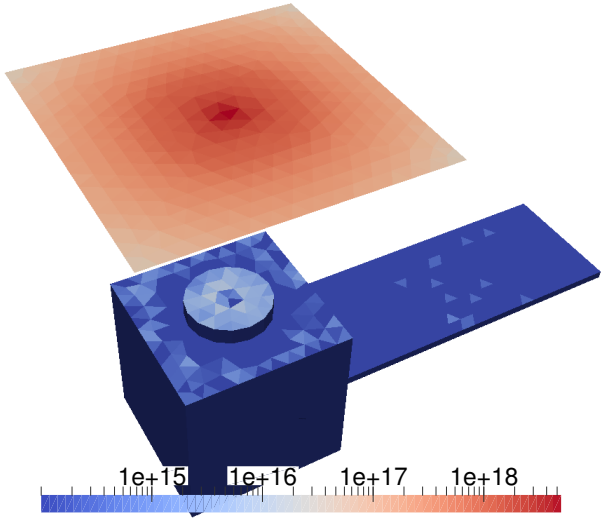
Figure 18. SMesh field data. The data are time-averaged, otherwise noted in the subfigure caption.



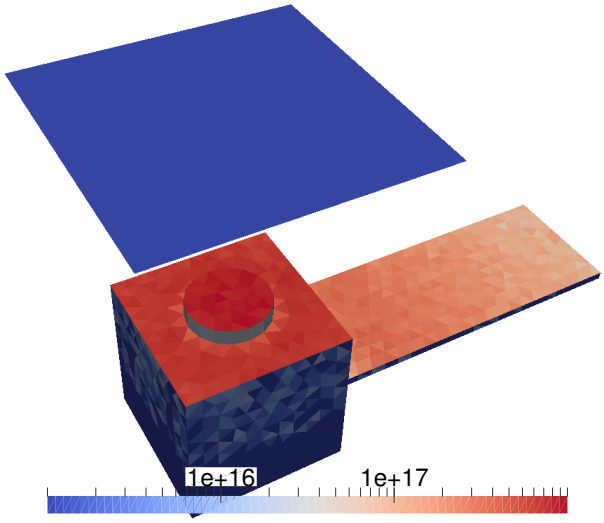
(a) Xenon atom



(b) Singly charged ion

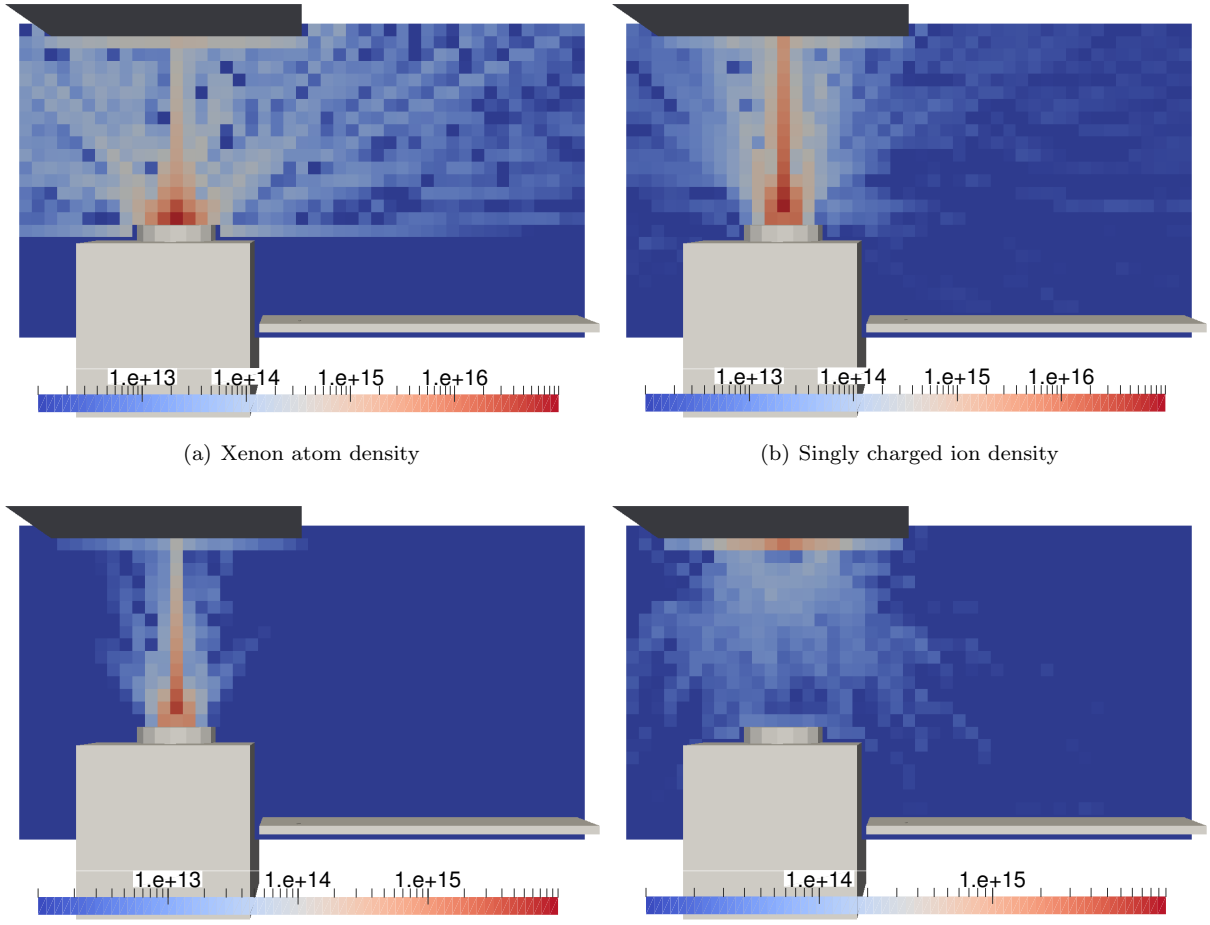


(c) Doubly charged ion



(d) Iron atom

Figure 19. Fluxes of different species to UMesh. All the data are time-averaged.



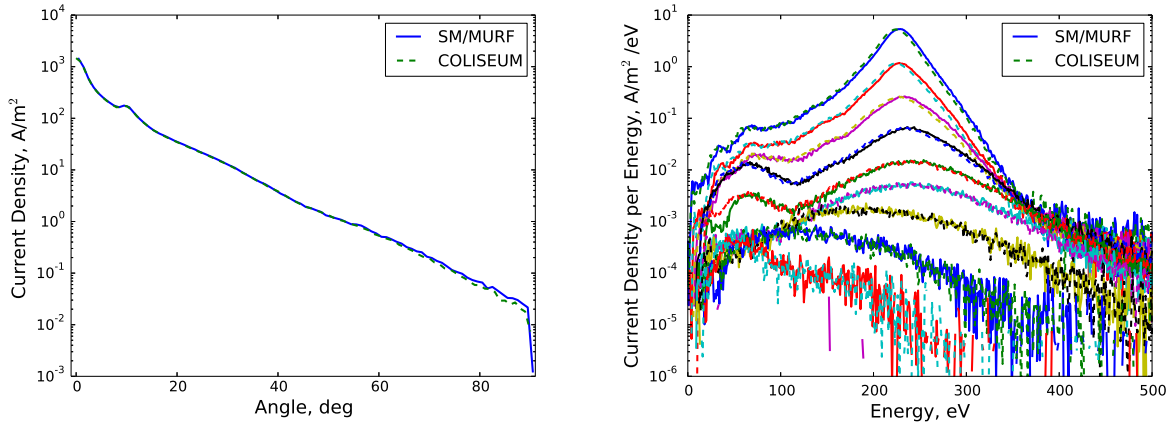
(a) Xenon atom density

(b) Singly charged ion density

(c) Doubly charged ion density

(d) Iron atom density

Figure 20. Differences in SMesh field data between SM/MURF and COLISEUM. The colormap scale is the same as in Fig. 18.



(a) Faraday probe.

(b) RPA probe.

Figure 21. Spherical stage probe data from SM/MURF and COLISEUM. Numerical measurements are taken at a spherical surface whose radius is about a third of distance from the Thruster face to Plate. The center of the spherical stage corresponds to the Thruster face center.

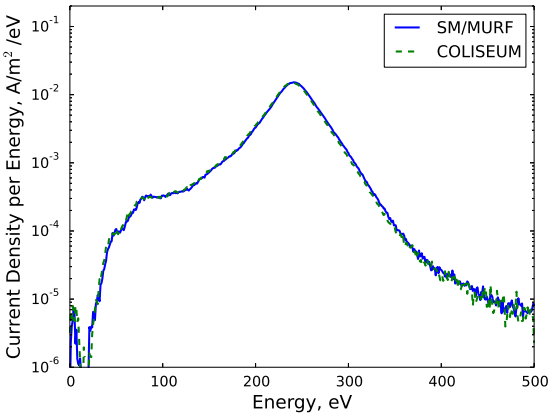
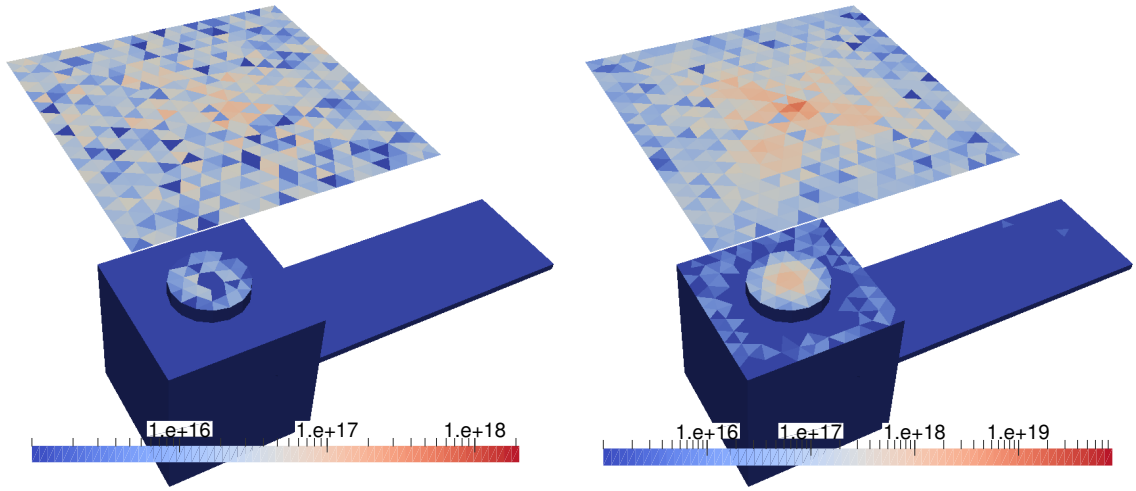
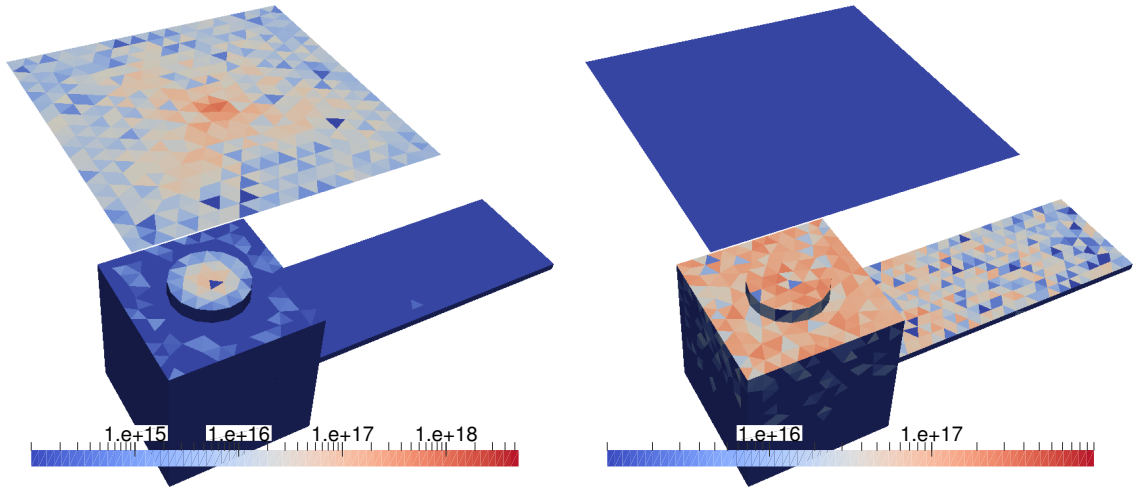


Figure 22. Fixed RPA probe data from SM/MURF and COLISEUM. The numerical probe is attached to Plate.



(a) Xenon atom

(b) Singly charged ion



(c) Doubly charged ion

(d) Iron atom

Figure 23. Differences in fluxes to the Plate surface between SM/MURF and COLISEUM. The colormap scale is the same as in Fig. 19.

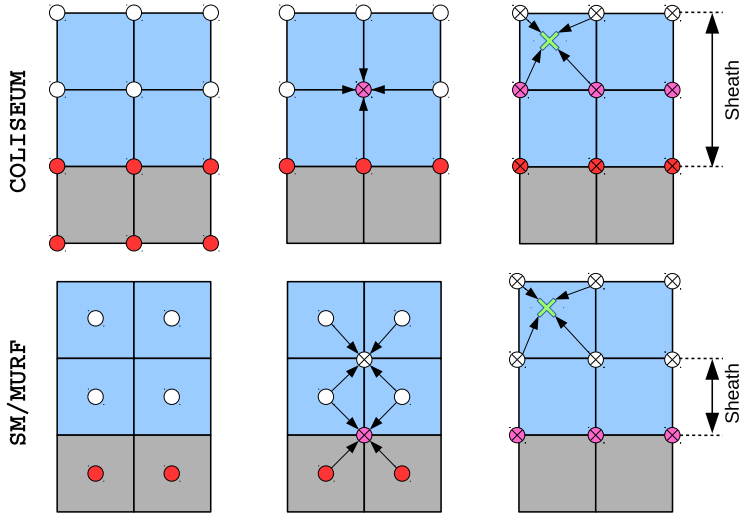


Figure 24. Illustration of electric field calculation near a geometry. Gray and blue cells are in a geometry or free-space, respectively. Electric potential and field are computed on open circles and circles with X, respectively. The sheaths are purely numerical.

Even with a very small difference in the probe data, it appears that the sputtered iron distribution on the spacecraft can be significantly different as shown in Fig. 23(d). As seen in Fig. 23(a), the difference in the xenon atom flux to the Plate surface is generally small. The maximum and L2 norms are 0.06 and 0.007, respectively. In other words, the maximum difference of the xenon atom flux is only 6 %. However, the differences are larger for the singly and doubly charged ions. The maximum differences for the two species are about 13%. This is directly related to the slightly higher velocity distribution in SM/MURF. Therefore, iron atoms are sputtered at higher rates. Furthermore, according to the sputtering model, larger incident energy generally leads to a larger sputter yield value, causing more particles to be sputtered off from the surface. The two factors cause a larger flux out the Plate surface, thus increasing the flux to the spacecraft. The maximum and L2 norms of the relative differences are 0.24 and 0.033, respectively.

Lastly, COLISEUM predicts much larger ion fluxes to the Thruster face compared to SM/MURF. This is primarily caused by the electric field calculation near a surface. Both COLISEUM and SM/MURF set electric potentials inside spacecraft components to be some user-specified values. While the component potentials are set to be exactly the same between SM/MURF and COLISEUM, the electric field near the spacecraft components are different as illustrated in Fig. 24. The Thruster face is aligned with nodes where COLISEUM set the potential to be some constant. The electric field on a node into the free space by one cell width is calculated with the component potential. Then, the electric field on a particle right above the node is computed using the field value at the node. Since the component potential can affect the electric field on a particle located less than two cell width from the Thruster surface, the numerical sheath extends two cell wide in COLISEUM for this specific simulation set-up. On the other hand, due to the potential calculation on cell-centered locations in SM/MURF, the numerical sheath extends one cell wide. As a result, slow ions are more likely to be pulled toward the wall in COLISEUM. The difference between the two codes should be minimized if the Thruster face is aligned with the cell-centers, resulting in the numerical sheath of 1.5 cell width in both SM/MURF and COLISEUM.

VI. Conclusion

While the SM/MURF framework is still being actively developed, SM/MURF is now capable of performing full plume simulations for flight support. Up to date, COLISEUM has been used for such plume integration simulations, but SM/MURF is intended to be much more flexible and expandable. Every operation required for plume simulations has been tested by unit and/or integration tests. In order to make sure that the current capabilities are not broken by future updates, these tests are set up to run regularly on our server through a continuous integration tool, Jenkins. As the final verification test, SM/MURF and COLISEUM are used to perform the same baseline plume simulation, and the results from the two codes are compared. In general, the agreement between the two codes is very good. A slight discrepancy is caused by

where the field data are computed; SM/MURF and COLISEUM compute field data on cell-centers and nodes, respectively. This results in a slightly different electric field along the thruster axis, which cascades through particle trajectories, field calculations, sputter rate, and redeposition rate onto spacecraft components.

Currently SM/MURF and COLISEUM take about the same amount of time to complete a simulation. The next release version of SM/MURF will focus on applying new algorithmic techniques to increase its numerical efficiency. Several techniques are being included, such as: (i) sub-cycling of faster species to reduce the number of steps for slow particles, (ii) sub-cycling of HPHall code to use a larger time-step in SM/MURF, (iii) fast elastic collision calculation using a lookup table and effective elastic cross-section,[29] (iv) particle merge and split algorithm to limit the number of particles while improving resolution in the region of interest.,[30] and (v) a MPI/GPU hybrid plume simulation.

References

[1]Martin, R. and Koo, J., *Thermophysics Universal Research Framework - Infrastructure Release*, Air Force Research Laboratory (AFRL/RQRS), Version 1.0.0 ed., Apr 2015.

[2]Fife, J. M., Gibbons, M. R., Hargus, W. A., Van Gilder, D. B., Kirtley, D. E., and Johnson, L. K., "3-D Computation of Surface Sputtering and Redeposition due to Hall Thruster Plumes," *28th International Electric Propulsion Conference*, Toulouse, France, 2003, pp. 1–7, IEPC 2003–0136.

[3]Cheng, S., Santi, M., Celik, M., Martinez-Sanchez, M., and Peraire, J., "Hybrid PIC-DSMC Simulation of a Hall Thruster Plume on Unstructured Grids," *Computer Physics Communications*, Vol. 164, 2004, pp. 73–79.

[4]Brieda, L., Kafafy, R., Pierru, J., and Wang, J., "Development of the DRACO Code for Modeling Electric Propulsion Plume Interactions," *40th AIAA/ASME/SAE/ASEE Joint Propulsion Conference and Exhibit*, Fort Lauderdale, Florida, 2004, pp. 1–21, AIAA 2004-3633.

[5]Brieda, L., *Multiscale Modeling of Hall Thrusters*, Ph.D. thesis, George Washington University, Washington D. C., 2012.

[6]Bird, G. A., *The DSMC Method*, CreateSpace Independent Publishing Platform, 2013.

[7]Box, G. E. P. and Muller, M. E., "A Note on the Generation of Random Normal Deviates," *Annals of Mathematical Statics*, Vol. 29, No. 2, 1958, pp. 610–611.

[8]Bird, G. A., *Molecular Gas Dynamics and the Direct Simulation of Gas Flows*, Clarendon Press, Oxford, 1994.

[9]Fife, J. M., *Hybrid-PIC modeling and electrostatic probe survey of Hall thruster*, Ph.D. thesis, Massachusetts Institute of Technology, Cambridge, MA, 1998.

[10]Celik, M., Santi, M., Cheng, S., Martinez-Sanchez, M., and Peraire, J., "Hybrid-PIC Simulation of a Hall Thruster Plume on an Unstructured Grid with DSMC Collisions," *28th International Electric Propulsion Conference*, Toulouse, France, 2003, pp. 1–10, IEPC 2003–134.

[11]Santi, M., Celik, M., Cheng, S., Martinez-Sanchez, M., and Peraire, J., "Further Development and Preliminary Results of the AQUILA Hall Thruster Plume Model," *39th AIAA/ASME/SAE/ASEE Joint Propulsion Conference and Exhibit*, Huntsville, Alabama, 2003, pp. 1–11, AIAA 2003–4873.

[12]Brieda, L. and Keidar, M., "Multiscale Modeling of Hall Thrusters," *32nd International Electric Propulsion Conference*, Kurhaus, Wiesbaden, Germany, 2011, pp. 1–13, IEPC 2011–101.

[13]Möller, T. and Trumbore, B., "Fast, Minimum Storage Ray/Triangle Intersection," *Journal of Graphics Tools*, Vol. 2, No. 1, 1997, pp. 21–28.

[14]Greenwood, J., "The correct and incorrect generation of a cosine distribution of scattered particles for Monte-Carlo modelling of vacuum systems," *Vacuum*, Vol. 67, 2002, pp. 217–222.

[15]Matsunami, N., Yamamura, Y., Itikawa, Y., Itoh, N., Kazumata, Y., Miyagawa, S., Morita, K., Shimizu, R., and Tawara, H., "Energy dependence of the ion-induced sputtering yields of monatomic solids," *Nuclear Data Tables*, Vol. 31, 1984, pp. 1–80.

[16]Yamamura, Y. and Tawara, H., "Energy dependence of the ion-induced sputtering yields of monatomic solids at normal incidence," *Nuclear Data Tables*, Vol. 62, 1996, pp. 149–253.

[17]Kannenberg, K., Khayms, V., Emgushov, B., Werthman, L., and Pollard, J., "Validation of Hall Thruster Plume Sputter Model," *37th Joint Propulsion Conference*, Salt Lake City, Utah, July 2001, pp. 1–10, AIAA 2001-3986.

[18]Roussel, J.-F., Bernard, J., and Garnier, Y., "Numerical simulation of induced environment, sputtering, and contamination of satellite due to electric propulsion," *Proceedings Second European Spacecraft Propulsion Conference*, Aug. 1997, pp. 517–522.

[19]Garnier, Y., Viel, V., Roussel, J.-F., and Bernard, J., "Low-Energy Xenon Ion Sputtering of Ceramics Investigated for Stationary Plasma Thrusters," *Journal of Vacuum Science and Technology A*, Vol. 17, No. 6, Nov 1990, pp. 3246–3254.

[20]Pencil, E. J., Randolph, T., and Manzella, D., "End-of-life Stationary Plasma Thruster far-field plume characterization," *32nd Joint Propulsion Conference*, Lake Buena Vista, Florida, July 1996, pp. 1–28, AIAA 1996-2709.

[21]Zhang, Z. L. and L., Z., "Anisotropic Angular Distributions of Sputtered Atoms," *Radiation Effects and Defects in Solids*, Vol. 159, 2004, pp. 301–307.

[22]Yamamura, Y., "Contribution of Anisotropic Velocity Distribution of Recoil Atoms to Sputtering Yields and Angular Distributions of Sputtered Atoms," *Radiation Effects*, Vol. 55, No. 1-2, 1981, pp. 49–55.

[23]Yamamura, Y., "Theory of Sputtering and Comparison to Experimental Data," *Nuclear Instrument and Methods*, Vol. 194, 1982, pp. 515–522.

[24]Verboncoeur, J., "Particle Simulation of Plasmas: Review and Advances," *Plasma Physics and Controlled Fusion*, Vol. 47, 2005, pp. A231–A260.

[25]Miller, J. S., Pullins, S. H., Levandier, D. J., Chiu, Y.-H., and Dressler, R. A., "Xenon Charge Exchange Cross Sections for Electrostatic Thruster Models," *Journal of Applied Physics*, Vol. 91, No. 3, 2002, pp. 984–991.

[26]Scharfe, M. K., Koo, J. W., and Azarnia, G., "DSMC implementation of experimentally-based Xe^{+} + Xe differential cross sections for electric propulsion modeling," *AIP Conference Proceedings*, Vol. 1333, AIP, 2011, pp. 1085–1090.

[27]Giuliano, P. N. and Boyd, I. D., "Effects of Detailed Charge Exchange Interactions on DSMC-PIC Simulation of a Simplified Plasma Test Cell," *32nd International Electric Propulsion Conference*, Wiesbaden, Germany, 2011, pp. 1–10, IEPC 2011–112.

[28]Araki, S. J. and Wirz, R. E., "Ion-Neutral Collision Modeling Using Classical Scattering with Spin-Orbit Free Interaction Potential," *IEEE Transactions on Plasma Science*, Vol. 41, No. 3, 2013, pp. 470–480.

[29]Araki, S. J., "Fast Computation of High Energy Elastic Collision Scattering Angle for Electric Propulsion Plume Simulation," *30th International Symposium on Rarefied Gas Dynamics*, Victoria BC, Canada, 2016, pp. 1–9.

[30]Martin, R. S. and Cambier, J.-L., "Moment Preserving Adaptive Particle Weights using Octree Velocity Distributions for PIC Simulations," *AIP Conference Proceedings*, Vol. 1501, AIP, 2012, pp. 872–879.

SM/MURF: Current Capabilities and Verification as a Replacement of AFRL Plume Simulation Tool COLISEUM

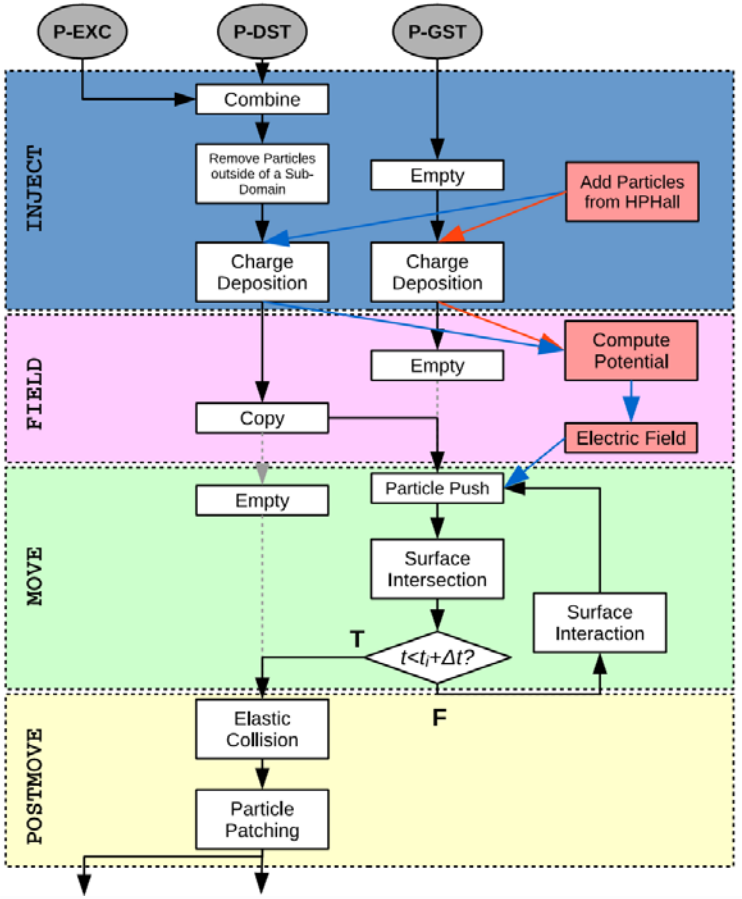
Samuel J. Araki[1], Robert S. Martin[1],
David L. Bilyeu[2], and Justin W. Koo[2]

ERC Inc[1], In-Space Propulsion Branch[2],
Air Force Research Laboratory
Edwards Air Force Base, CA USA

52nd AIAA/SAE/ASEE Joint Propulsion Conference,
Salt Lake City, July 25-27th, 2016

Outline

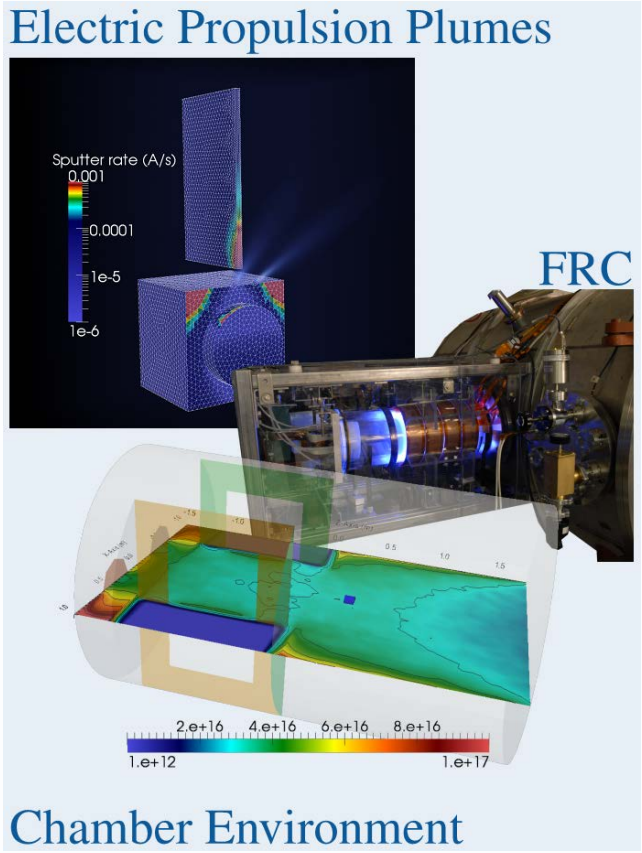
- Introduction
- SM/MURF Framework
- Operations for Plume Simulations
- Comparison with COLISEUM
- Conclusion/Future Work



Motivation / Objective

Motivation

- Multiple research codes from group
- Redundant functionality
- Unified framework across different models
- Support for GPU/OpenMP for accelerator/co-processor technology
  - COLISEUM: AFRL plume simulation tool developed since early 2000s – Only MPI support with domain decomposition strategy



Objectives

- First release version of SM/MURF with COLISEUM capability
- Verification of operations through unit and integration tests
- Final verification with a baseline plume simulation

Spacecraft Multi-Scale/Multi-Physics Universal Research Framework (SM/MURF)

Spacecraft Multi-Physics / Multi-Scale Universal Research Framework

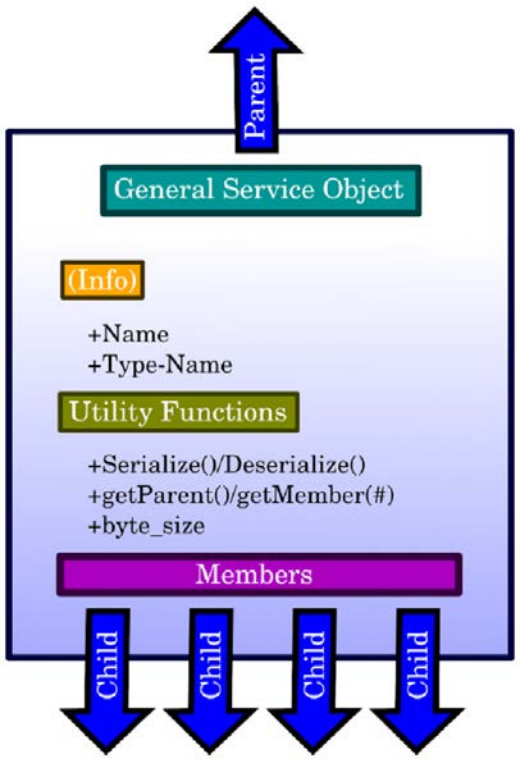
Thermophysics Universal Research Framework

- Thermophysics Universal Research Framework (TURF): Non-ITAR subset for collaborators
- SM/MURF includes an operation to communicate with a Hall thruster model (HPHall)

SM/MURF Framework

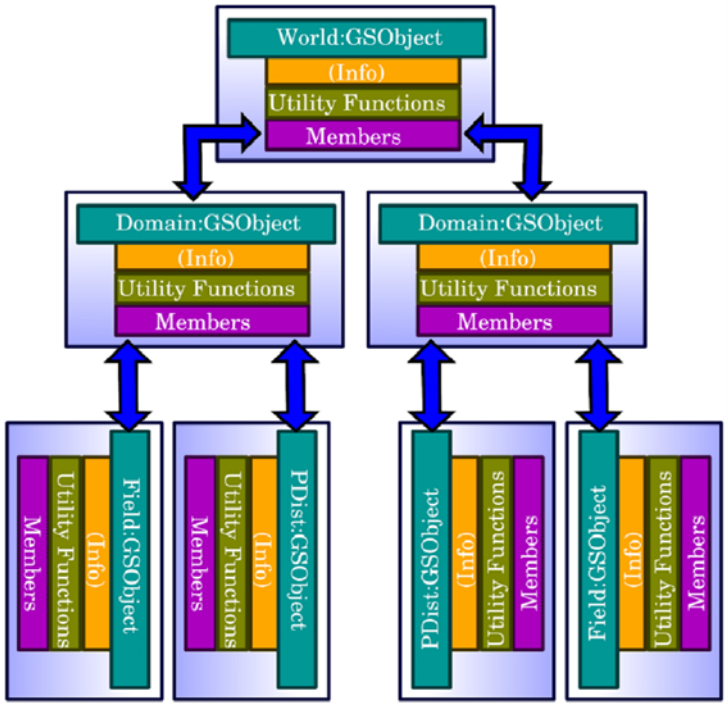
General Service Object (GSObject)

- Wraps every class object
- Unifies tree hierarchy



Hierarchical Tree-Structure

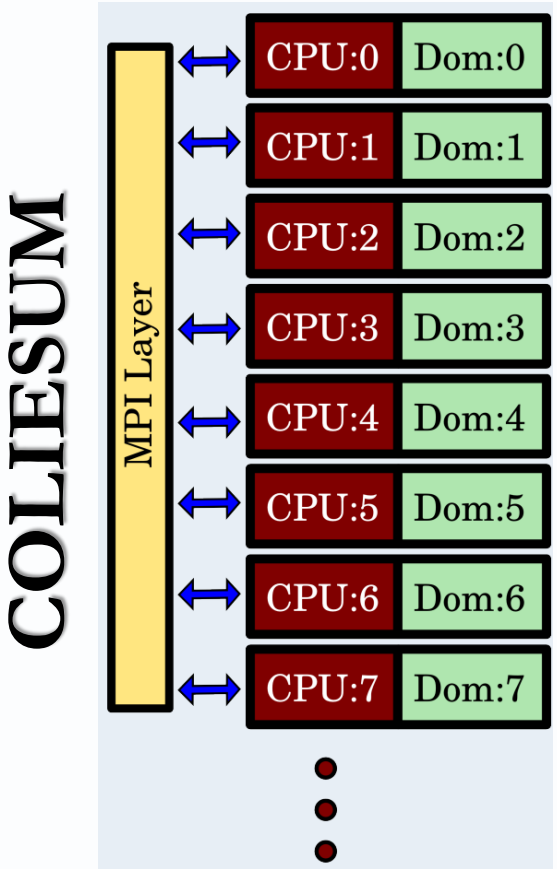
- Paths to other operations
- Enables recursive copy and structure traversal

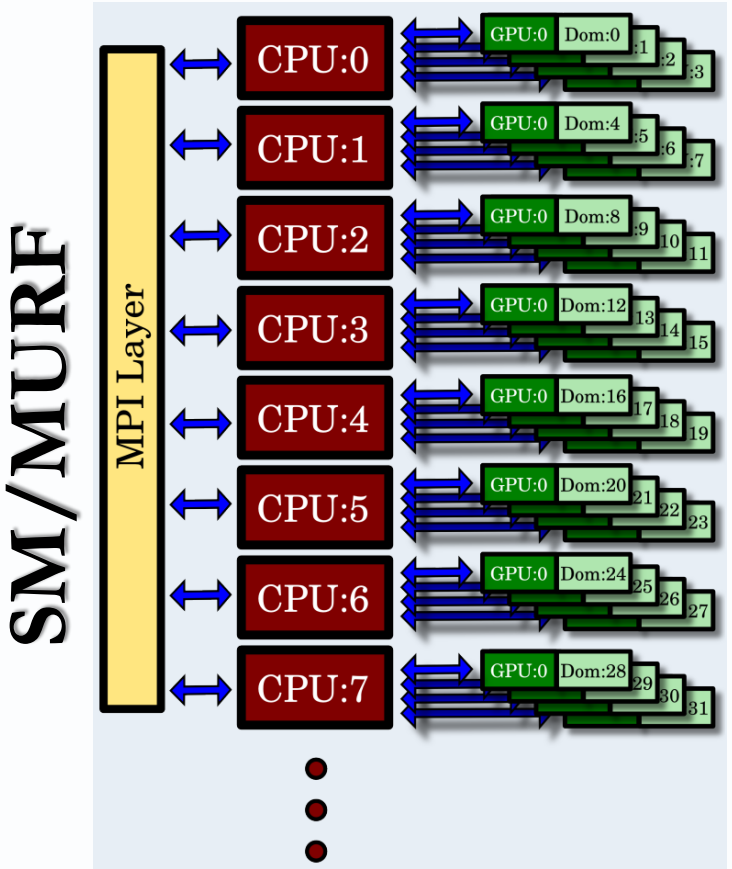


SM/MURF Framework

Multiple sub-domains per CPU

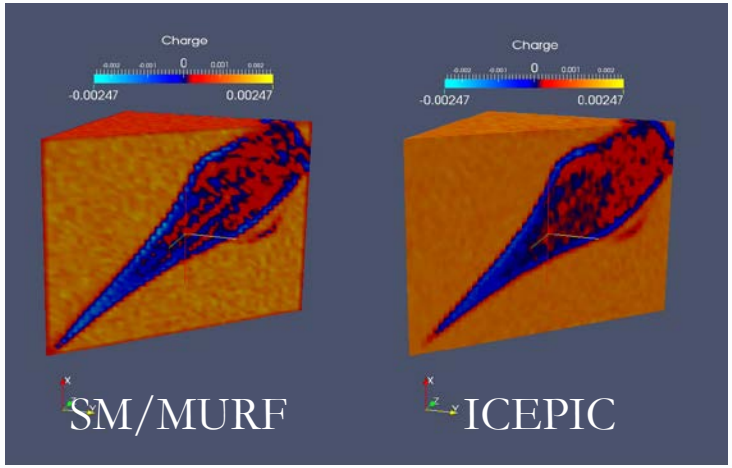
- Enables dynamic load balancing



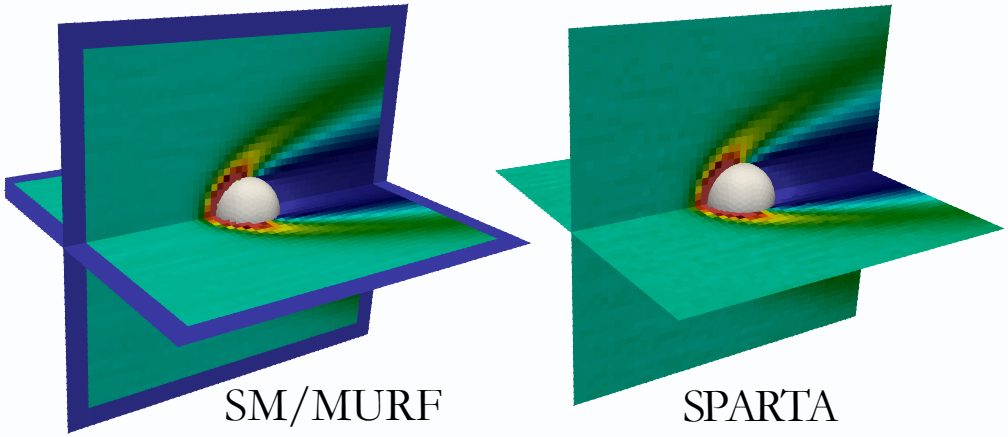


Examples of Results

Plasma in a Grounded Box

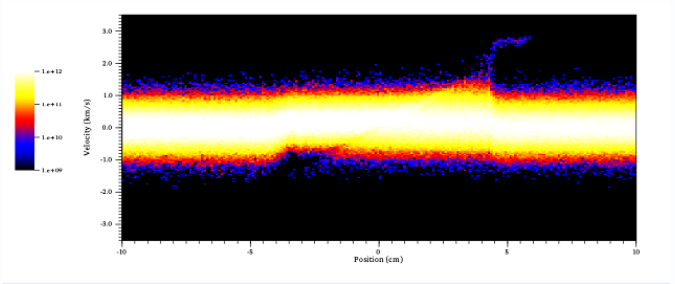


Supersonic Flow over a Sphere

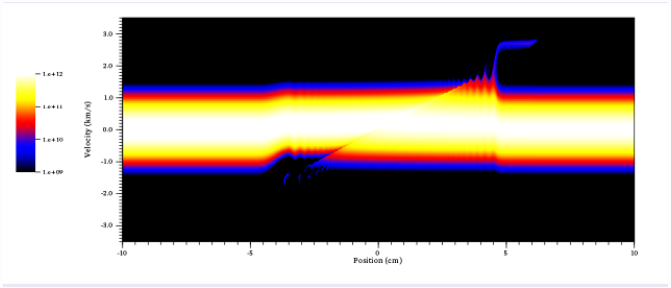


Collisionless Shock

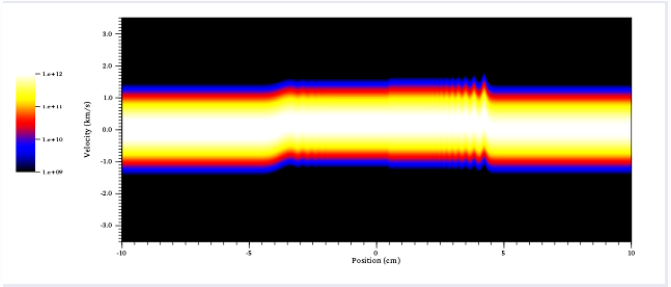
1D3V PIC



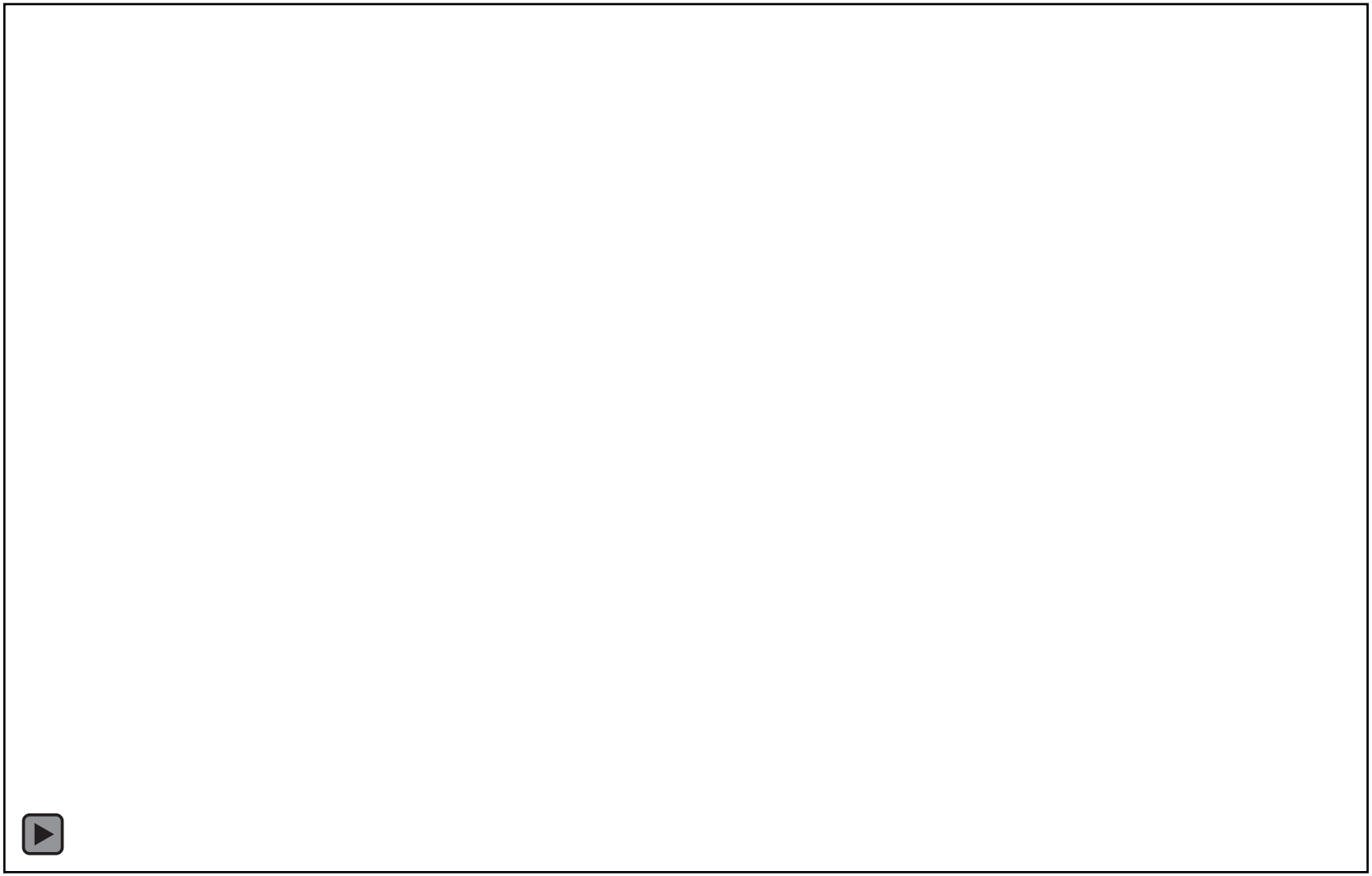
1D1V Vlasov



Fluid

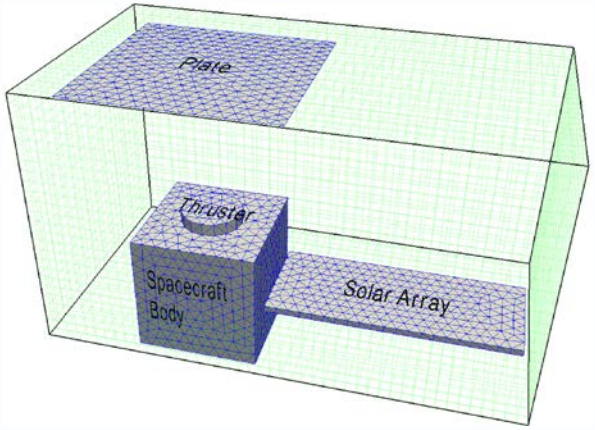


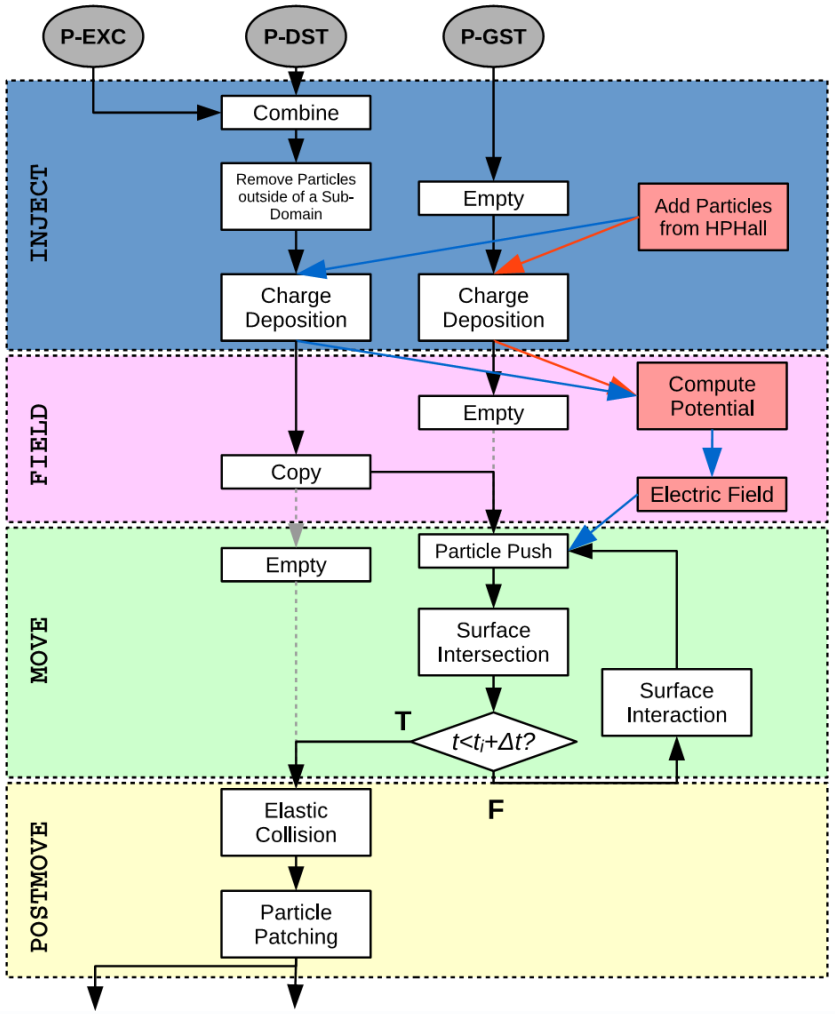
Collisionless Shock: PIC vs Vlasov



Baseline Plume Simulation

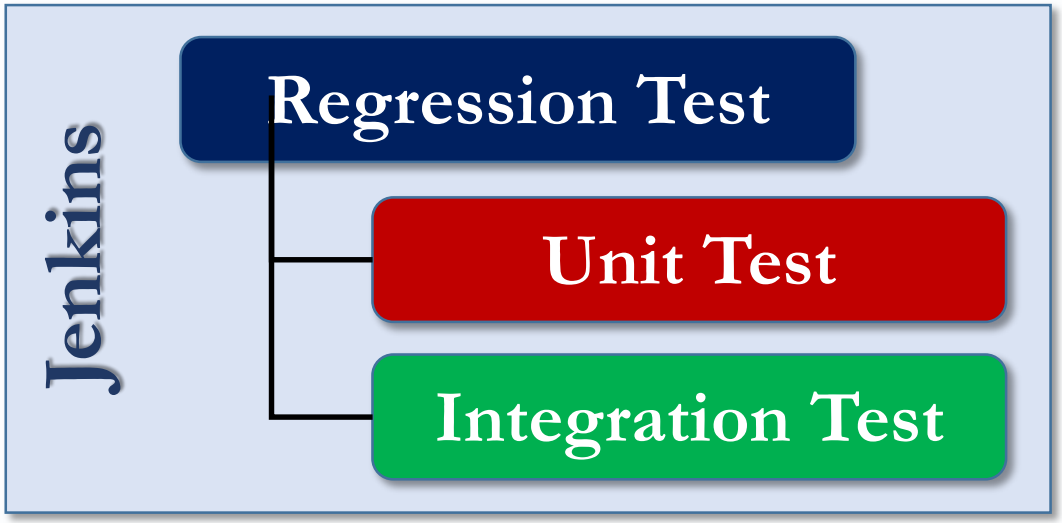
- PIC with heavy species only
  - Xe, Xe^{+}, Xe^{2+} from HPHall
  - Sputtered species from surfaces (BN, Fe, Al, C, etc)
- Electrons at Boltzmann equilibrium
- Four communication stages
  - `Inject`, `Field`, `Move`, and `Postmove`

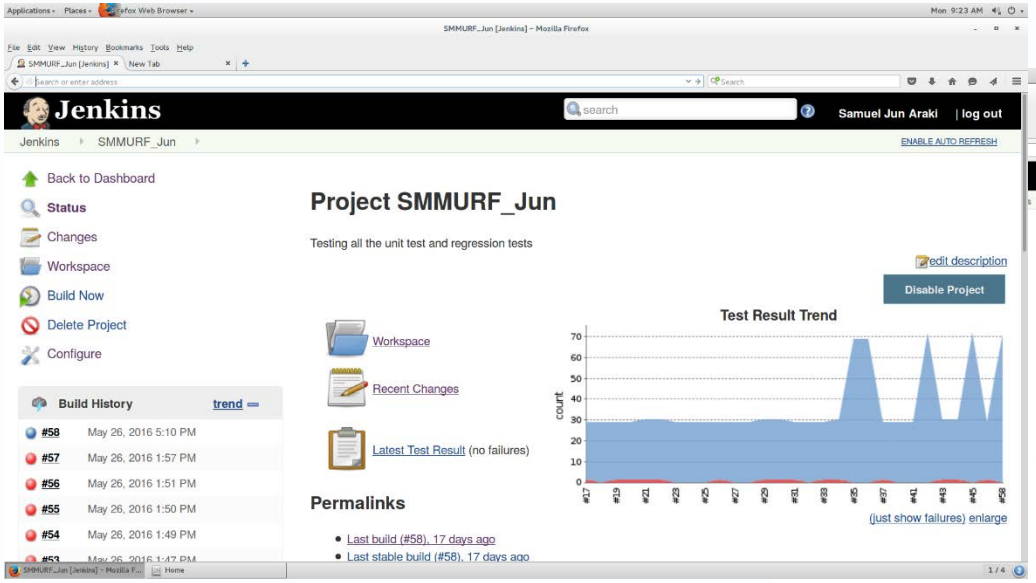


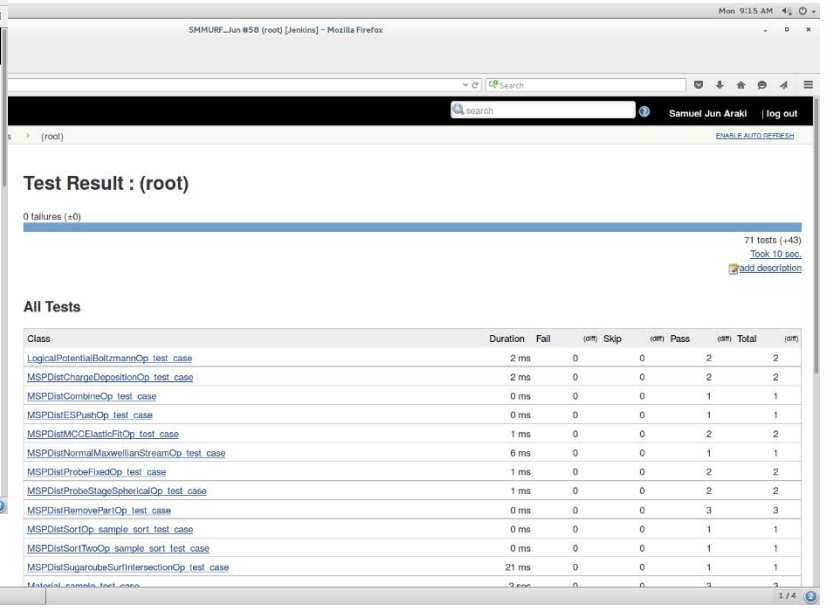


Regression Test with Jenkins

https://jenkins.io/

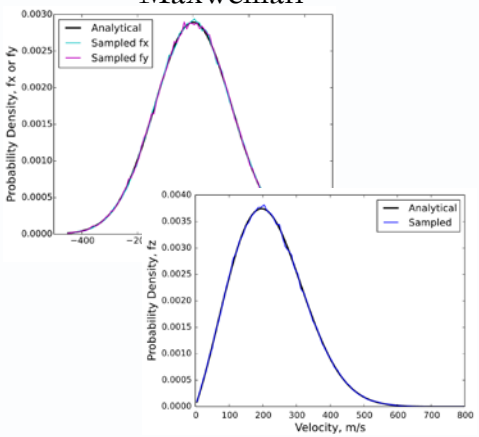




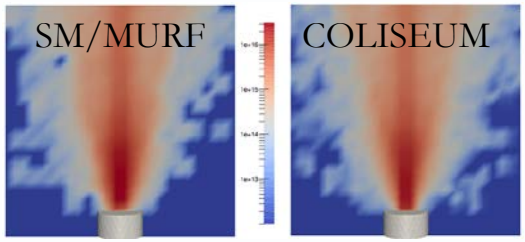


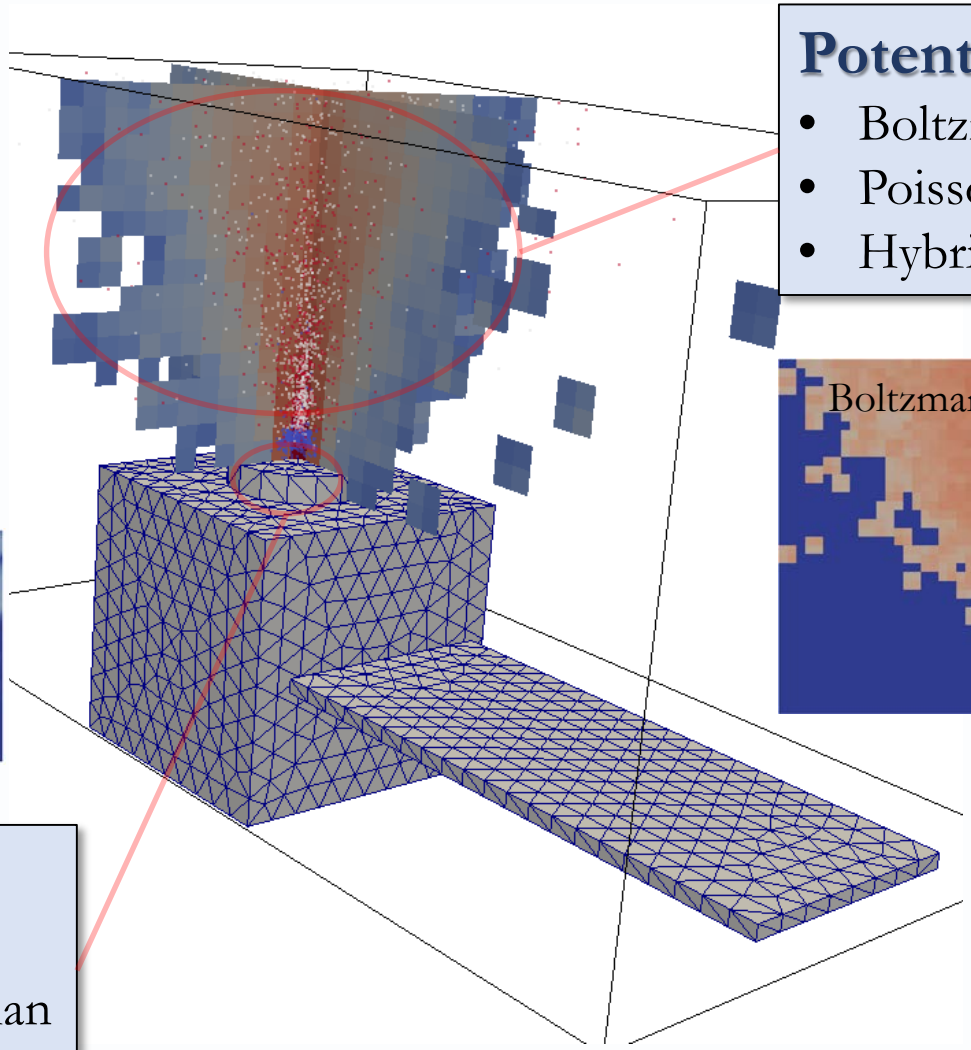
Source Model / Potential Solver

VDFs from Streaming Maxwellian



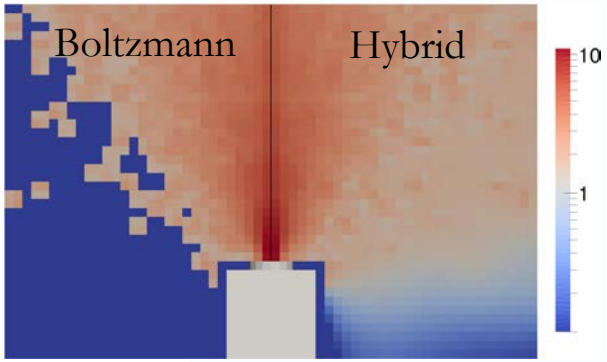
HPHall Source





Potential Solver

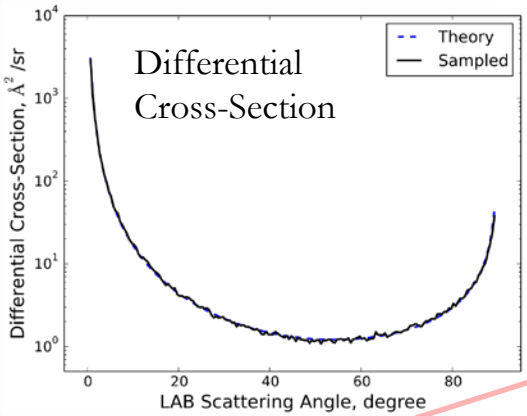
- Boltzmann inversion
- Poisson solve
- Hybrid Model

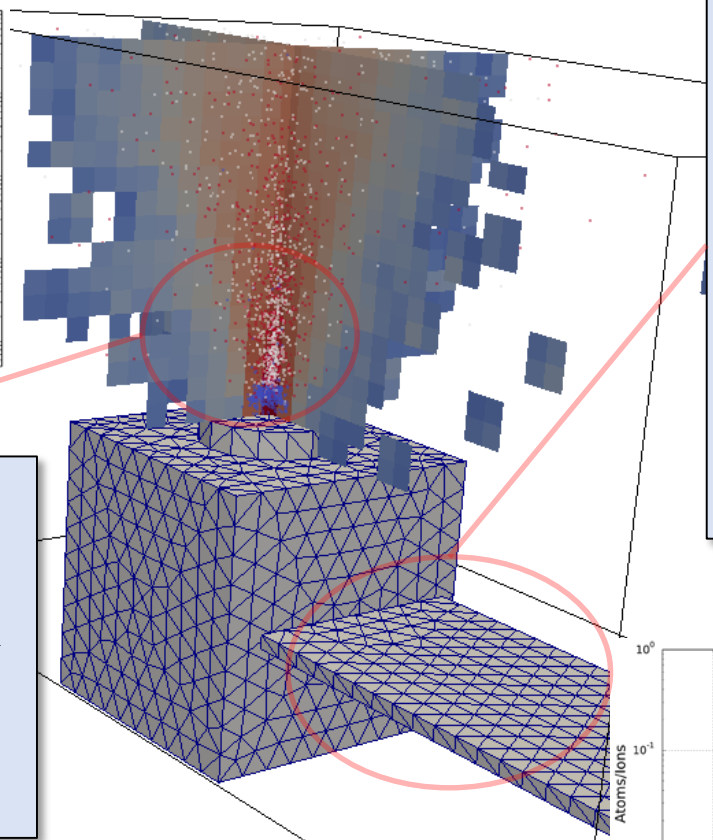


Source Models

- HPHall source
- Streaming Maxwellian
- RPA source

Collision / Surface Interaction



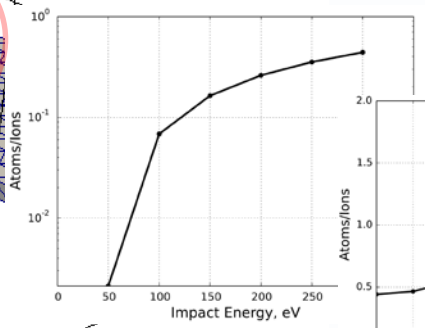


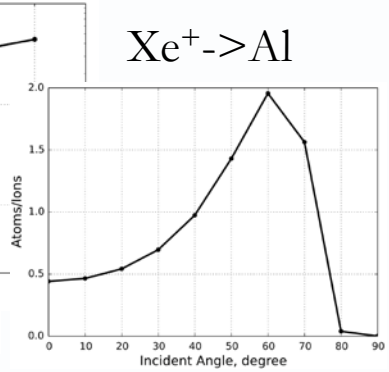
Surface Interactions

- Reflection
  - Specular/Diffuse
- Sputtering
  - Matsunami et al.[2]
  - Yamamura and Tawara[3]
  - Kannenberg et al.[4]
  - Roussel et al.[5]
  - Garnier et al.[6]
  - Pencil et al.[7]

Elastic Collision

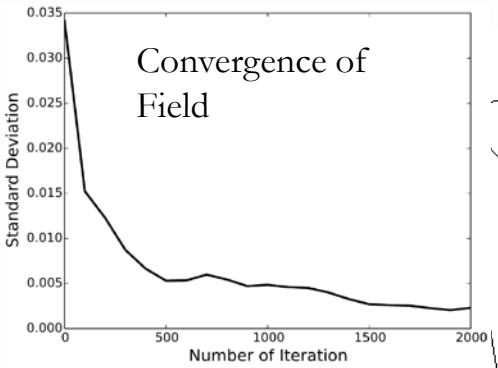
- MCC collision
  - Sample from differential cross-section
  - Classical scattering with spin-orbit free potential[1]

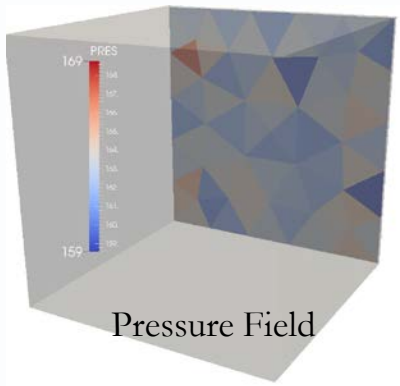


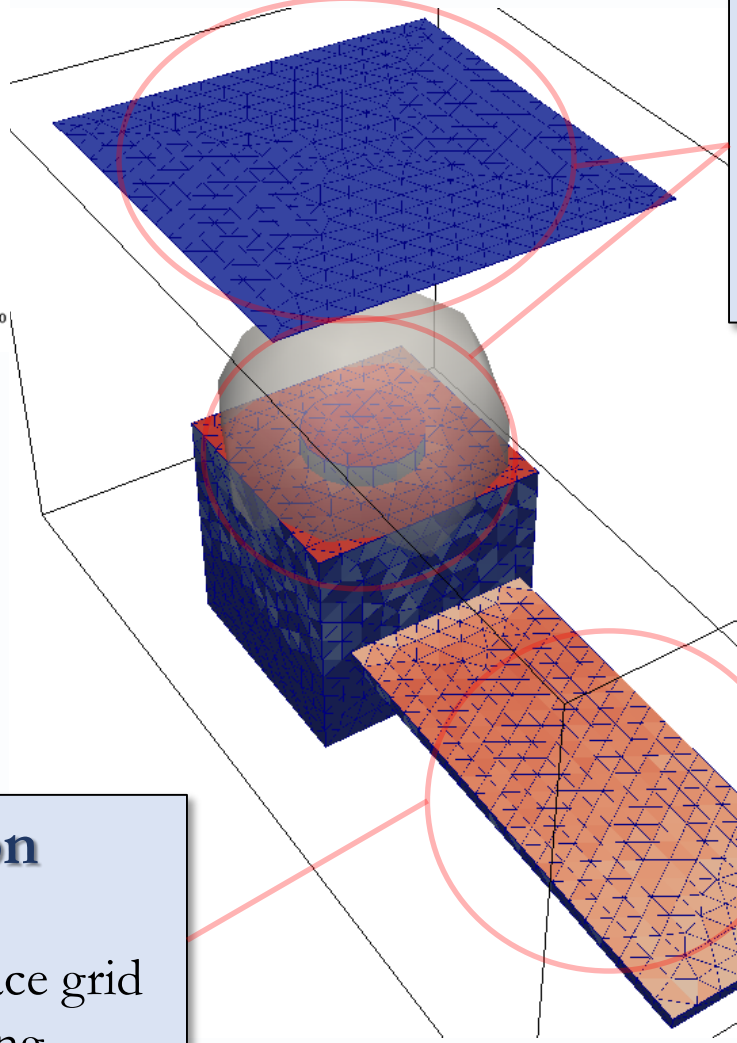


[1] Araki, S., *30th International Symposium on Rarefied Gas Dynamics*, Victoria BC, Canada, 2016, pp.
[2] Matsunami, et al, *Nuclear Data Tables*, Vol. 31, 1984, pp. 1–80.
[3] Yamamura, Y. and Tawara, H., *Nuclear Data Tables*, Vol. 62, 1996, pp. 149–253.
[4] Kannenberg, K. et al., *37th JPC*, Salt Lake City, Utah, July 2001, pp. 1–10, AIAA 2001-3986.
[5] Roussel, J.-F., Bernard, J., and Garnier, Y., *Proceedings Second European Spacecraft Propulsion Conference*, Aug. 1997, pp. 517–522.
[6]Garnier, Y., Viel, V., Roussel, J.-F., and Bernard, J., *Journal of Vacuum Science and Technology A*, Vol. 17, No. 6, Nov 1990, pp. 3246–3254.
[7] Pencil, E. J., Randolph, T., and Manzella, D., *32nd JPC*, Lake Buena Vista, Florida, July 1996, pp. 1–28, AIAA 1996-2709.

Field Calculation / Numerical Probe

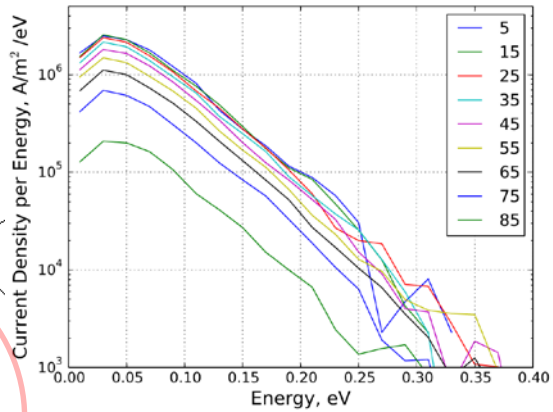






Numerical Probe

- Attached to a surface or a virtual probe with spherical stage
- Faraday or RPA



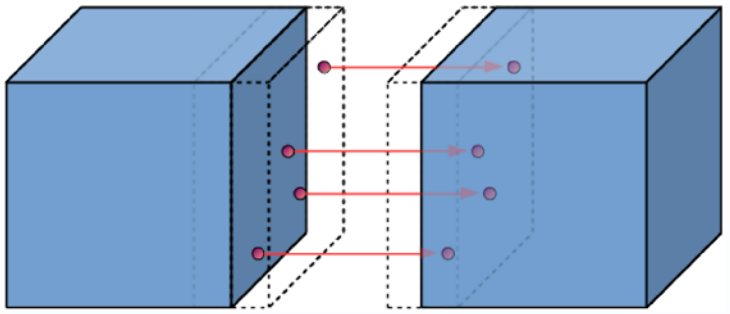
RPA Result from Maxwellian Source

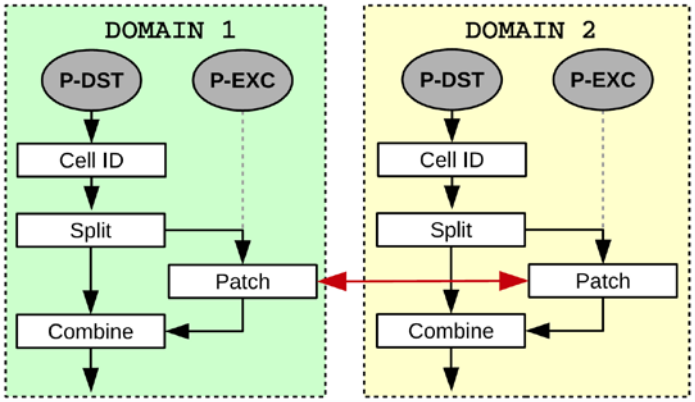
Field Calculation

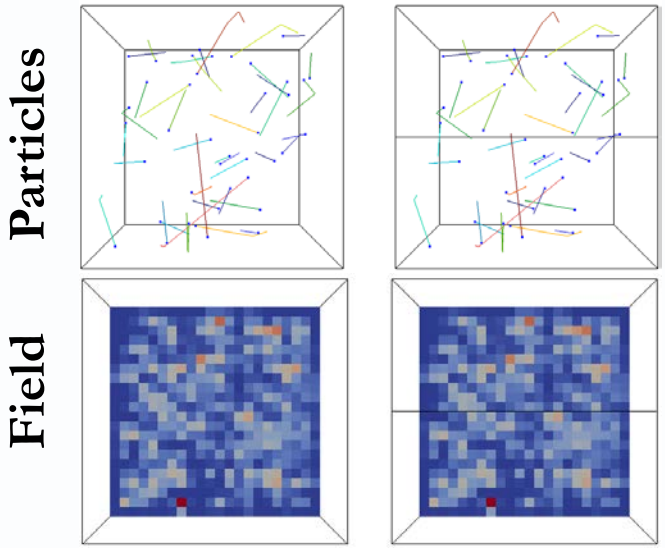
- Structured grid
- Unstructured surface grid
- DSMC-like sampling

Other Features

- **Particle and field patches across sub-domains**
  - Patch with ghost cells
  - MPI communications at the end of every stage



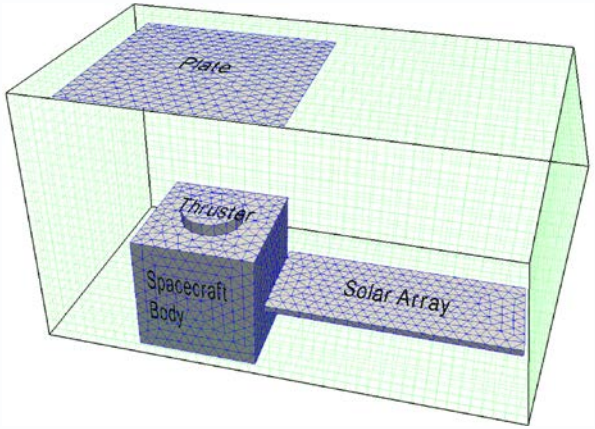


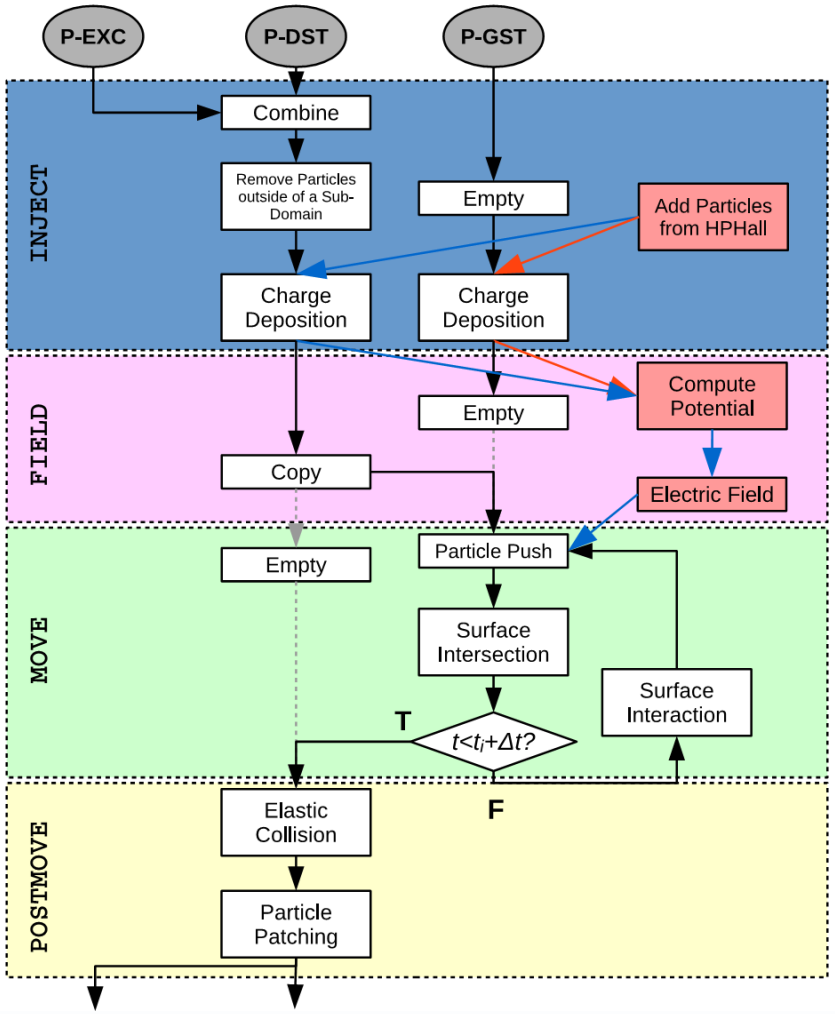


- **Particle and sampling field restart**
  - Particle: each sub-domain reads all the restart files
  - Field: each sub-domain reads one restart file

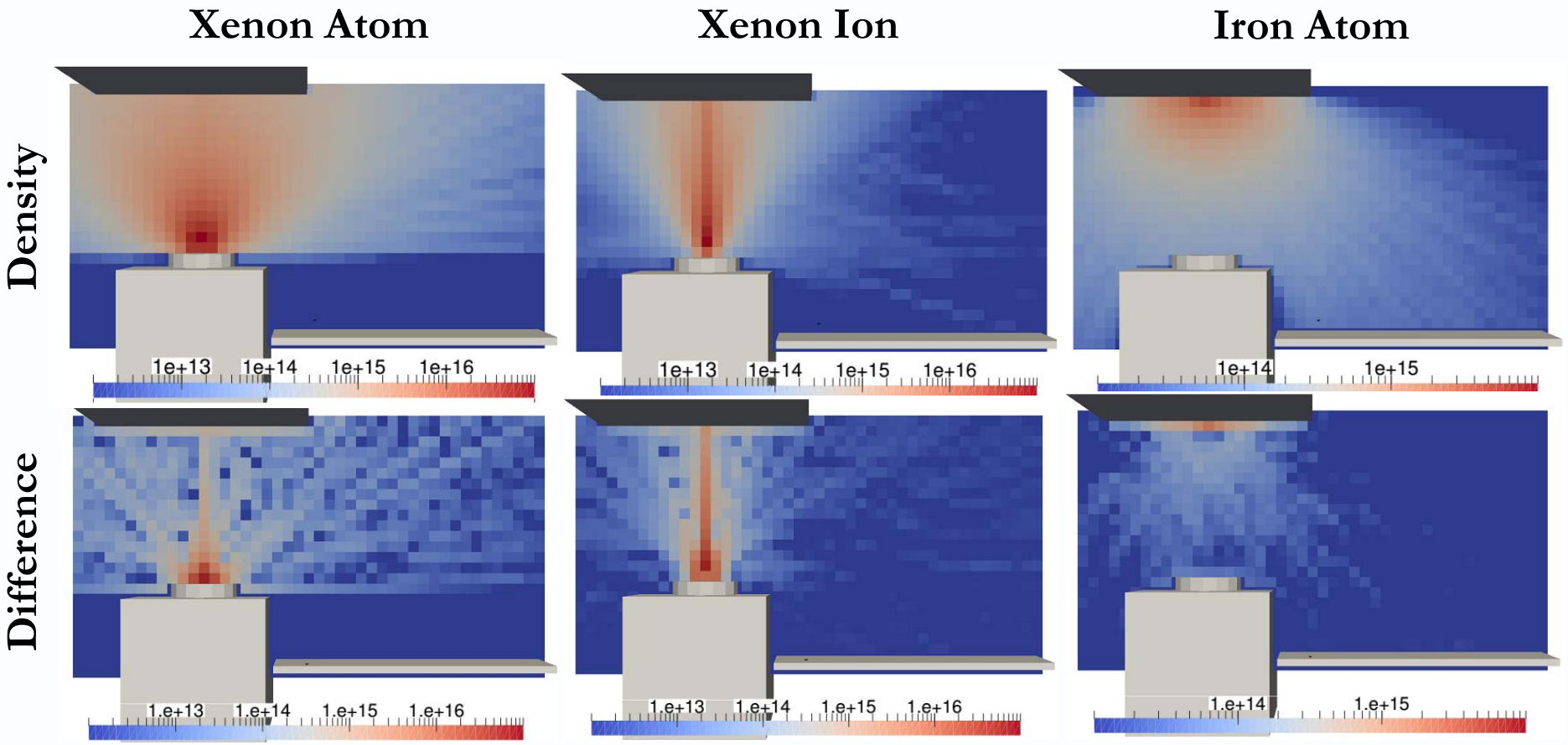
Baseline Plume Simulation

- PIC with heavy species only
  - Xe, Xe^{+}, Xe^{2+} from HPHall
  - Sputtered species from surfaces (BN, Fe, Al, C, etc)
- Electrons at Boltzmann equilibrium
- Four communication stages
  - `Inject`, `Field`, `Move`, and `Postmove`





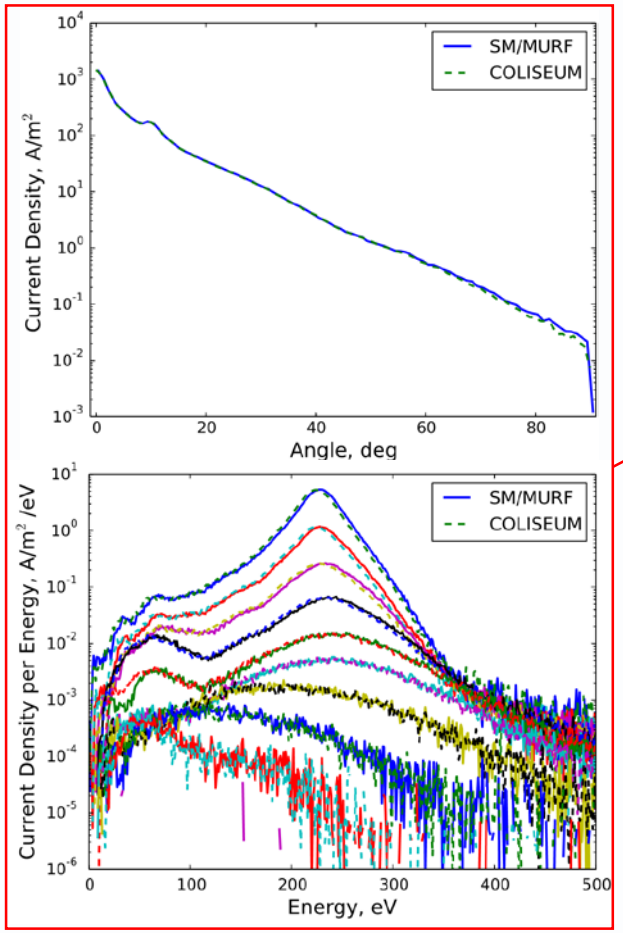
Comparison with COLISEUM: Field Data on Structured Grid

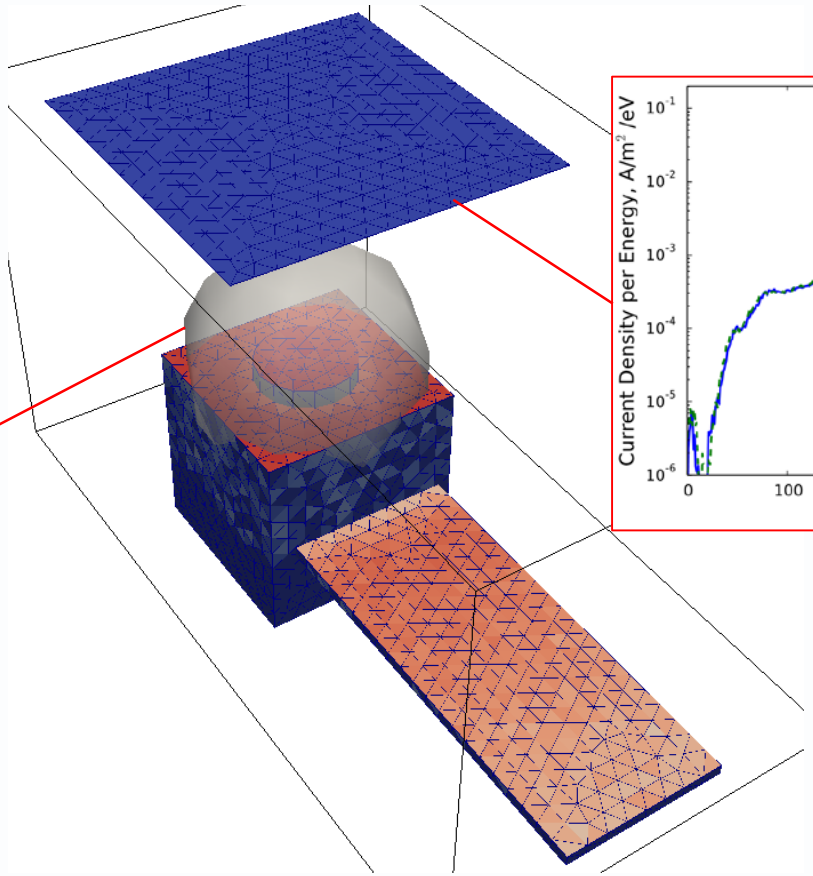


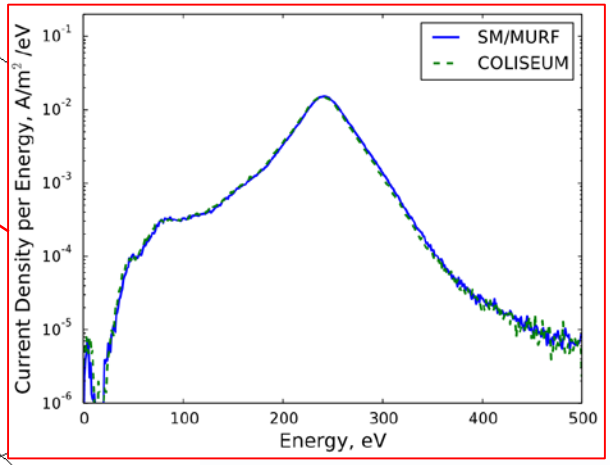
Cell-centered (SM/MURF) vs Node-centered (COLISEUM)

- Cells along Thruster center axis and near surface boundaries

Comparison with COLISEUM: Numerical Probes



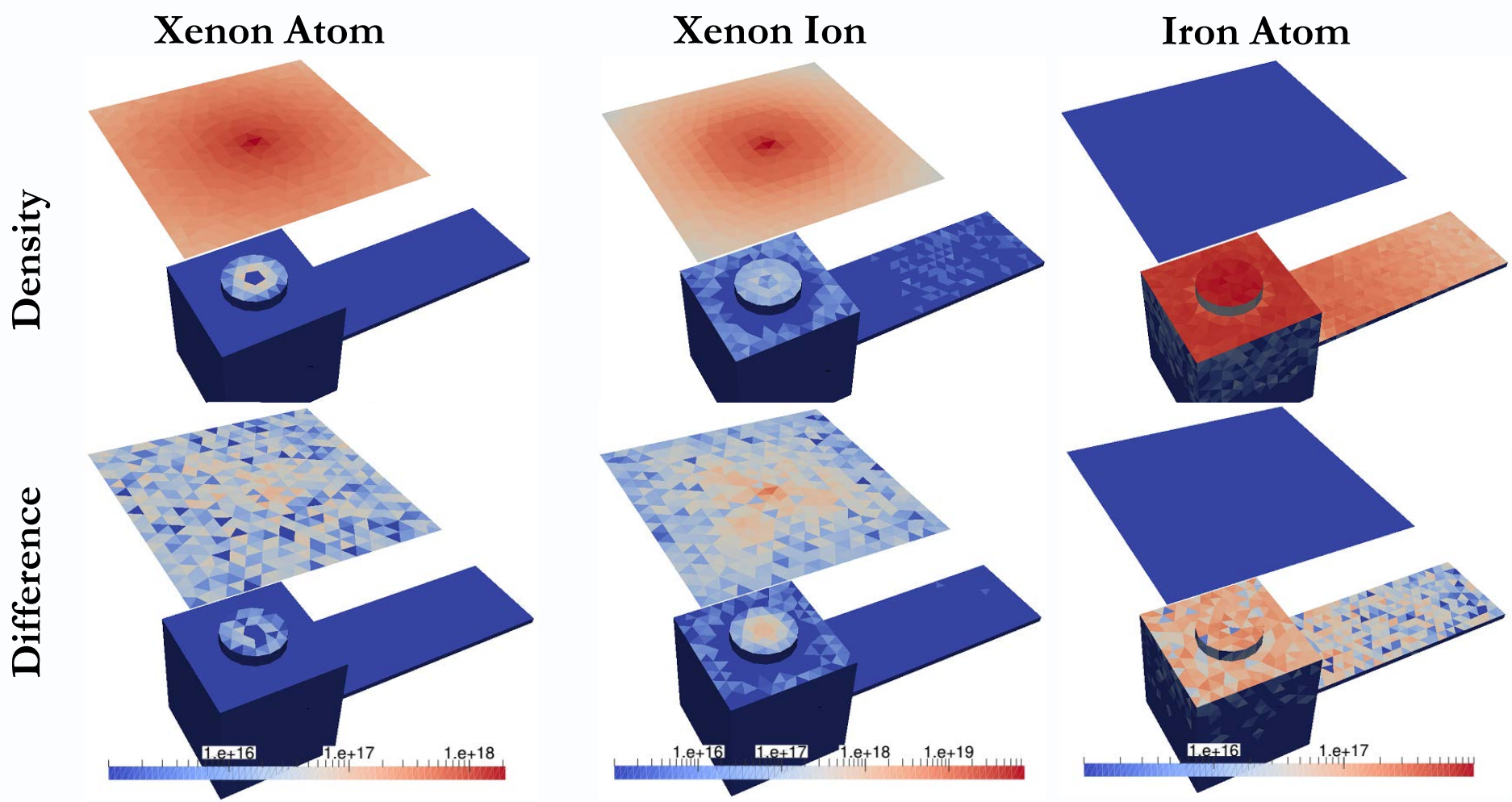




Excellent agreement between the two models

- Slight shift of SM/MURF data to higher energies

Comparison with COLISEUM: Field Data on Unstructured Grid



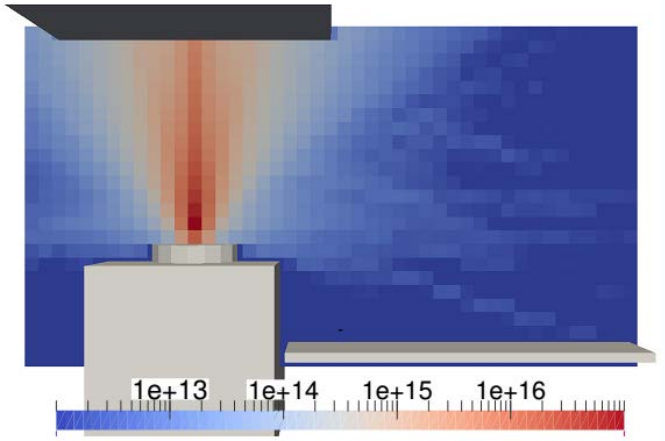
Difference in electric field cascades through other operations

- **Higher sputter and redeposition rate in SM/MURF**

Conclusion / Future Work

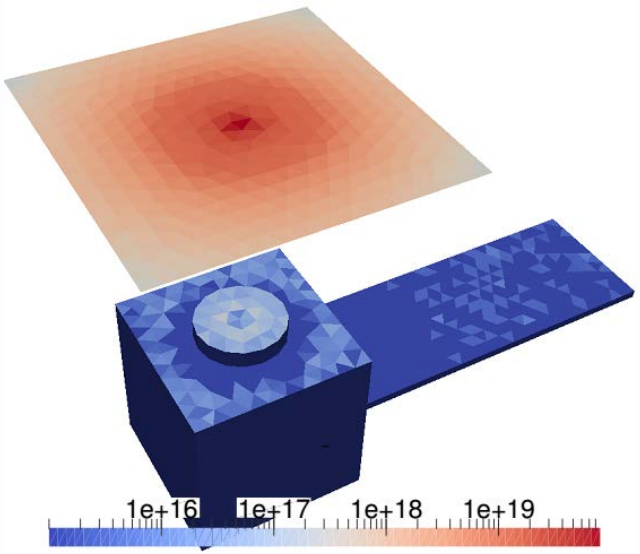
Conclusion

- First release version of SM/MURF is ready
- Operations for plume simulations are verified through multiple level unit and integration tests
- Regression tests with Jenkins for future development
- Good agreement with COLISEUM



Future Work

- Sub-cycling of fast species and HPHall
- Particle merge and split[1]
- Integration of Kokkos and MPI+GPU simulations
- Additional physics: spacecraft charging, multi-fluid with discontinuous Galerkin method, CR model, etc



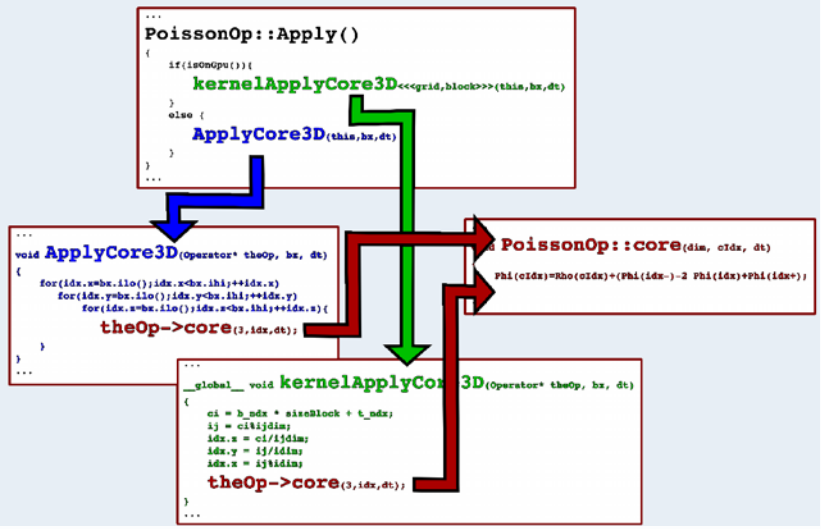
[1] Martin, R. S. and Cambier, J.-L., *AIP Conference Proceedings*, Vol. 1501, AIP, 2012, pp. 872–879

Thank you

SM/MURF Framework

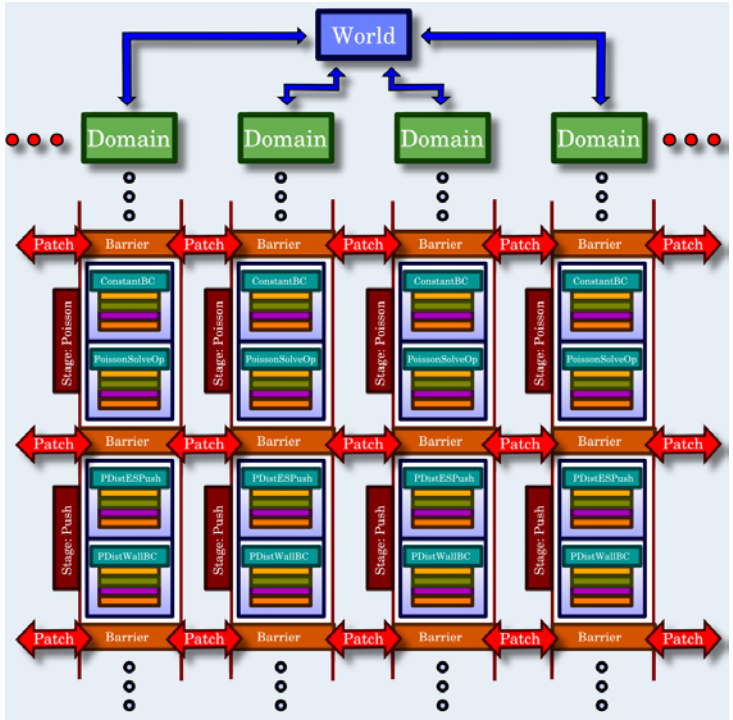
CPU/GPU Polymorphism

- Parallel region called by Core()
- Enables GPU and CPU routes to the same code

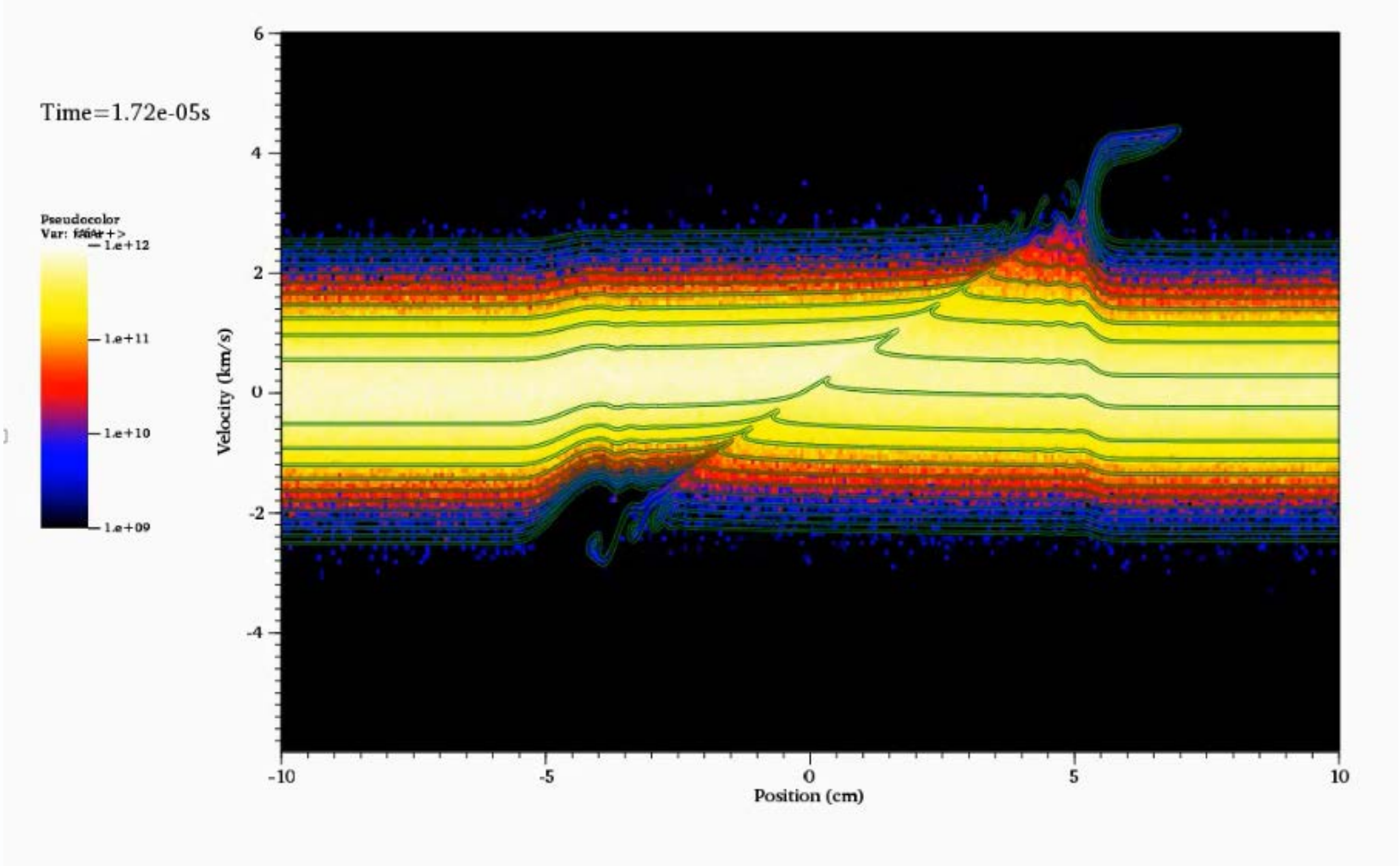


Operator Stacks

- Stacks executed in stages between communication

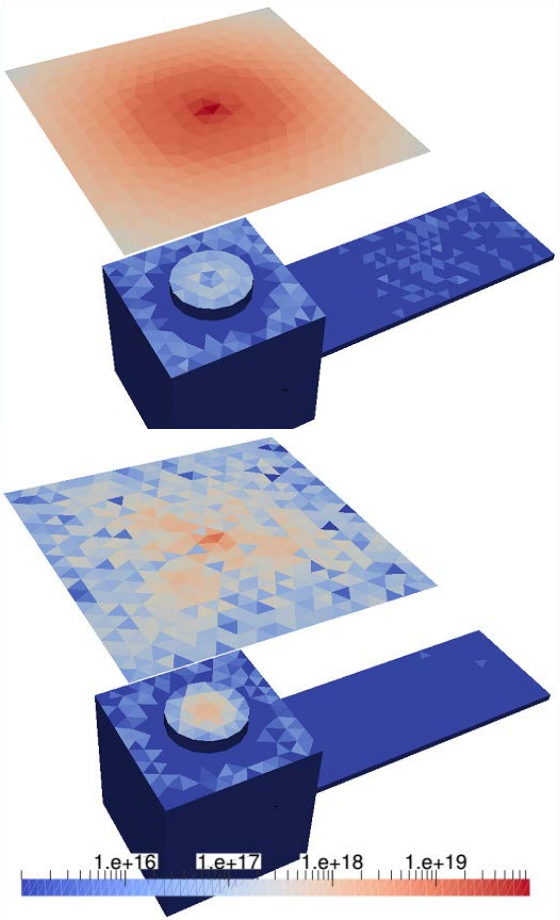


Collisionless Shock: PIC vs Vlasov



Numerical Sheath

Xenon Ion



Why different flux on Thruster face?

Difference in field calculation causes different sheath length